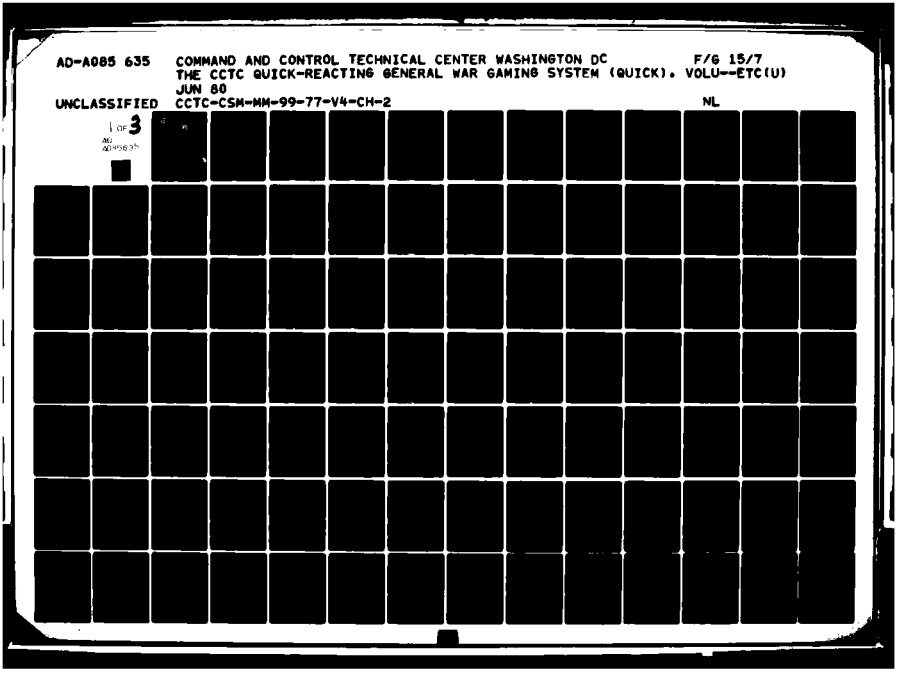
AD-A085 635 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 15/7
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK). VOLU--ETC(U)
JUN 80
UNCLASSIFIED CCTC-CSM-MM-99-77-V4-CH-2 NL



DEFENSE COMMUNICATIONS AGENCY
COMMAND AND CONTROL
TECHINCAL CENTER
WASHINGTON, D.C. 20301

12

LEVEL III

IN REPLY
REFER TO: C314

3 June 1980

TO: RECIPIENTS

SUBJECT: Change 2 to Program Maintenance Manual CSM MM 9-77, Volume IV, Sortie Generation Subsystem

1. Insert the enclosed change pages and destroy the replaced pages according to applicable security regulations.

2. Also enclosed is page 345 of change 1. It was originally printed incorrectly.

3. A list of Effective Pages to verify the accuracy of this manual is enclosed. This list should be inserted before the title page.

4. When this change has been posted, make an entry in the Record of Changes.

FOR THE DIRECTOR:

194 Enclosures
Change 2 Pages

J. DOUGLAS POTTER
Assistant to the Director
for Administration

(6) The CCTC Quick-Reacting General War Gaming System (QUICK). Volume IV. Sortie Generation Subsystem. Parts I and II. Program Maintenance Manual. Change 2.

-MM-9 -V4-CH-2

EFFECTIVE PAGES - APRIL 1980

This list is used to verify the accuracy of CSM MM 9-77, Volume IV after change 2 pages have been inserted. Original pages are indicated by the letter 0, change 1 pages by the numeral 1, and change 2 pages by the numeral 2.

Page No.	Change No.	Page No.	Change No.
Title Page, Part I	0	215	1
ii	2	216-260	0
iii-viii	2	260.1-260.2	2
ix	0	Title Page, Part II	0
1-1.2	2	ii-iii	2
2	2	iv	1
3	0	v	2
4-6	1	vi	1
7	0	vii-viii	0
8	1	261-265	0
9-10	0	266	1
11	2	266.1-266.2	1
12-18	0	267	0
19	1	268-269	1
20-21	0	269.1-269.2	1
22	1	270-271	0
23-26	0	272-273	1
27	2	274-293.2	2
28-63	0	294	2
64-65	2	295	1
66-69	0	296-299	0
70	2	300	2
71-97	0	301	0
98	2	302-307	2
99-104	0	308-309	0
105	1	310-311	2
106-109	0	312-317	0
110-110.30	2	318	2
111-129	0	319	0
130	2	320	2
131-132	0	321-322	0
133	2	323	2
134-143	0	324	1
144	2	325-326	0
145-209	0	327-328	1
210	2	329	2
211-213	0	330-331	0
214	2		

Page No.	Change No.	Page No.	Change No.
332-334	2	548	0
335-337	0	549	2
338-339	2	550-556	0
340	0	557-558	2
341-344	2	559-573	0
345-347	1	574	2
348-350	2	575-577	0
351	1	578-579	1
352	2	580-658	0
353	1	659	2
354-355	2	660-662	0
356	1		
357-361	2		
362-363	1		
364	2		
365	1		
366-367	2		
368	1		
369	2		
370-372	1		
373-376.2	2		
376.3-376.4	1		
376.5	2		
376.6	1		
376.7	2		
376.8	1		
377-409	0		
410	2		
411-414	0		
415	2		
416-462	0		
463	2		
464-486	0		
487	2		
488-497	0		
498-499	2		
500	0		
501-502	2		
503-509	0		
510	2		
511-538	0		
539	2		
540-542	0		
543-544	1		
545	2		
546	0		
547	2		

ACKNOWLEDGMENT

This document was prepared under the direction of the Chief for Military Studies and Analyses, CCTC, in response to a requirement of the Studies, Analysis, and Gaming Agency, Organization of the Joint Chiefs of Staff. Technical support was provided by System Sciences, Incorporated under Contract Number DCA 100-75-C-0019. *Change set one* was prepared under Contract Number DCA 100-78-C-0035. Computer Sciences Corporation prepared *change set two* under Contract Number DCA 100-78-C-0042.

CONTENTS

Part I

Section Page

ILLUSTRATIONS (PART I)

TABLES (PART I)

SECTION 1. GENERAL

1.1 Purpose

This volume of the QUICK Program Maintenance Manual describes the modules which are part of the QUICK Sortie Generation Subsystem. The information contained herein is presented on a module-by-module basis, complemented with discussions on computer programming maintenance on a subject-by-subject basis. The module-by-module discussions are structured so that a maintenance programmer can understand the program functions and programming techniques. The computer subjects are structured to inform the maintenance programmer of overall system programming techniques and conventions.

1.2 General Description

The Sortie Generation Subsystem operates using the integrated data base as developed by the Weapon Allocation subsystem and produces detailed bomber and missile (delivery vehicle and weapon) sortie specifications. Thus, it accepts a near-optimal weapon allocation, and from this as well as consideration of delivery vehicle characteristics and other factors. generates a detailed plan of attack for one opposing side in a hypothetical general war.

The subsystem consists of modules FOOTPRNT, MIRVDUMP, POSTALOC, PLANOUT and PLOTIT as shown in figure 1. Figure 2 shows the relationship of the Sortie Generation subsystem to other QUICK subsystems in terms of procedural information flow.

In addition to the plan generation requirements, per se, the output of this subsystem is utilized alternatively by:

a. Damage Assessment systems external to QUICK which utilize weapon/target strike data (DGZ tapes) as required.

b. General War simulation models external to QUICK (e.g., NEMO and ESP) which utilize relevant strike data as required (DGZ and A/B tapes).

If any missile delivery vehicles exists within the data base, module FOOTPRNT must be executed. For single shot missile delivery system individual sorties are simply formatted; no other action is required. For MIRV weapon groups detailed reentry vehicle target point assignment which satisfy the various constraints are created.

MIRVDUMP processes footprints developed by FOOTPRNT. The module determines alternate impact points for weapons from the same MIRVed sortie with the same impact points.

POSTALOC processes bomber weapon groups and develops specific bomber sorties.

All weapon groups are processed only if the attack posture indicator (attribute ATTPOS) equals zero. Otherwise only those weapon groups are processed where attack (attribute ATTINC) equals ATTPOS.

THIS PAGE INTENTIONALLY LEFT BLANK

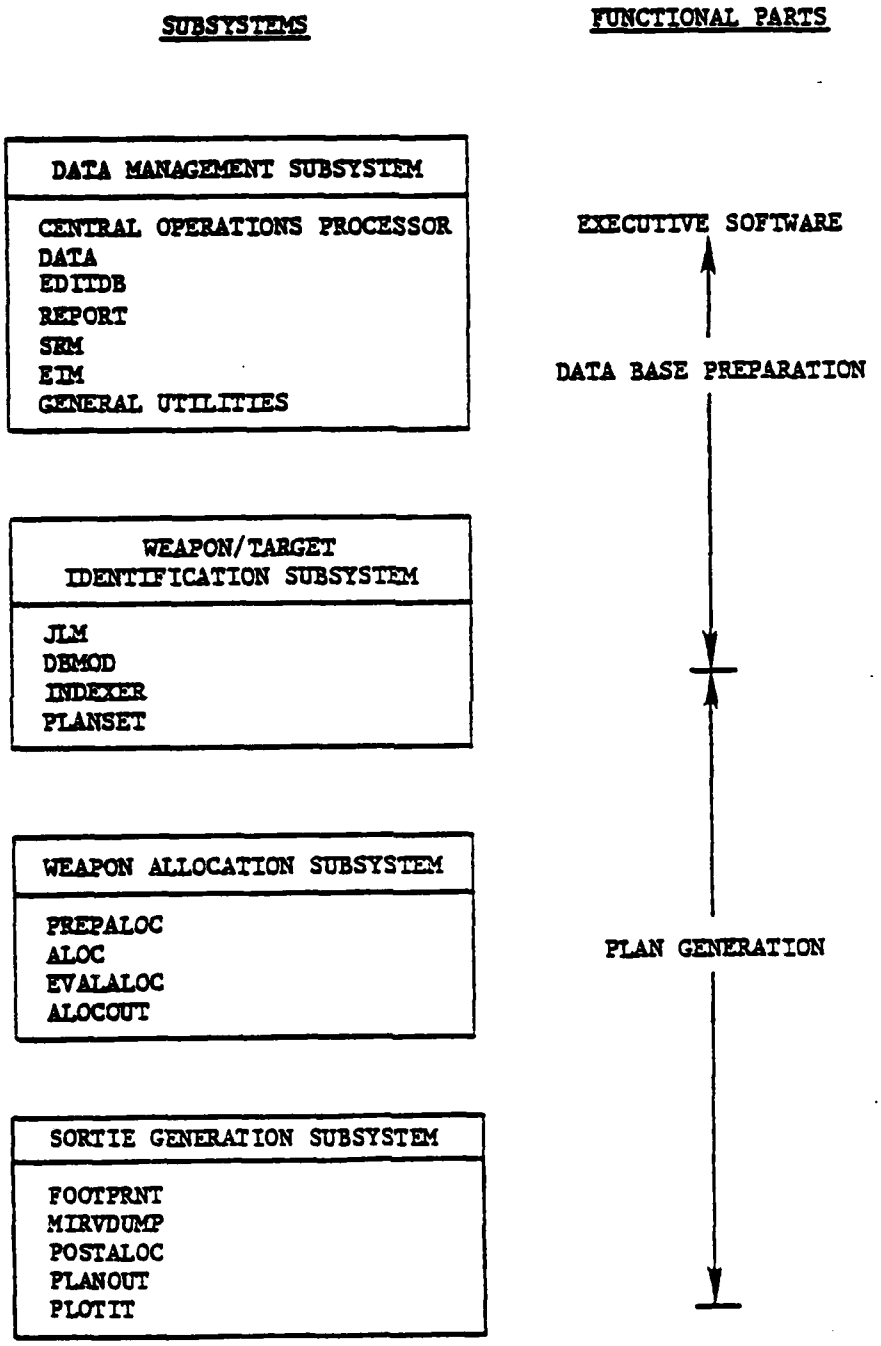


Figure 1. Major Subsystems of the QUICK System

Several subroutines develop the AOS form. These are:

- TABLINPT - Reads user requests and stores converted equations within the data base linked to the MIRV payload name.
- MKAOS - Called by TABLINPT to convert text English input into AOS form
- TAOS - Tests equations developed by MKAOS for errors
- PRAOS - Prints AOS developed equations
- XAOS - Executes AOS equations. Called by subroutine AXIS.

2.5.2 Subroutine DRIVER. This subroutine is called twice for each weapon group. DRIVER queries the strike list and collects a subset of targets that potentially may form a footprint. Subroutine PATHFIND is executed to ascertain if a footprint can be formed within the collected subset. If a footprint may be formed, DRIVER assigns strikes to given launch boosters and continues processing by collecting another subset of targets.

2.5.3 Subroutine PATHFIND. Given a list of targets, subroutine PATHFIND employs the footprint constraints and attempts to find a path among the targets that will form a feasible footprint. PATHFIND uses a dynamic programming algorithm that is a variation of the classical traveling salesman problem.

2.5.4 Development of Missile Plans. Subroutine MISASGN carries out the assignment of specific strikes to specific delivery vehicles within a weapon group. Figure 5 illustrates the structure of a typical group that MISASGN is designed to handle. The group may include several squadrons (two shown) and a squadron may include several sites (four per squadron shown). Each site may have one or more vehicles (three shown). Vehicles are considered to occupy the same site if they are so close together that they would have to be targeted as a single element target. For example, the Polaris squadron of 16 missiles on one submarine is considered to occupy one site, while the Minuteman squadron of 50 missiles occupies 50 separate sites.

On the other hand, any nonalert missiles in a squadron will constitute a separate weapon group. Since the vehicle indices within a squadron may not start from one, the starting vehicle index ISTART for each squadron is supplied as an input to the missile assignment phase. This and the other input parameters defining the available weapons for the program are shown in the figure. The strikes assigned to the group by program ALOC are placed in order by decreasing values of RVAL. The strikes are then assigned in this order beginning with the vehicle index ISTART for the first squadron. The next strike goes to the next squadron until all squadrons have one strike assigned. Then the vehicle index is

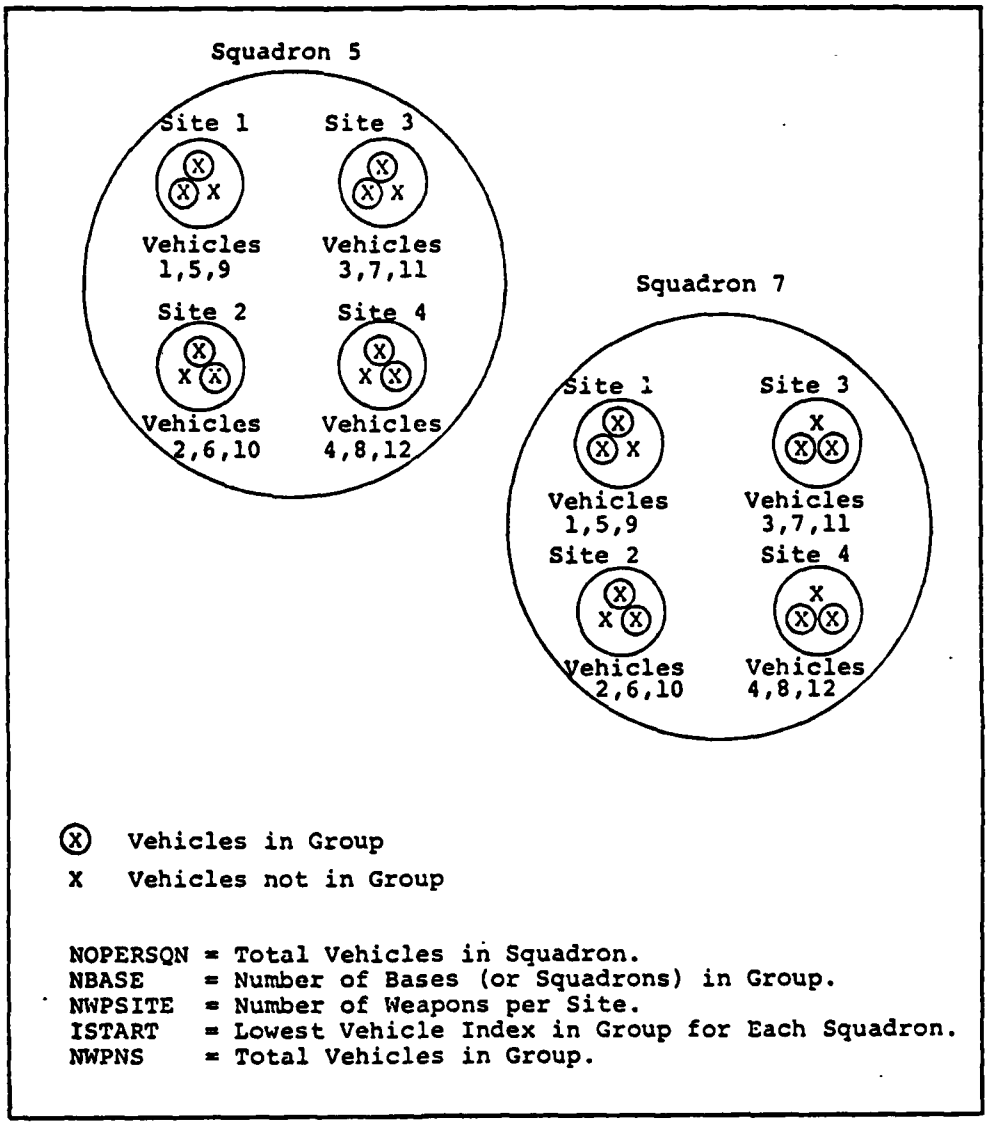


Figure 5. Configuration of Missiles in a Typical Group

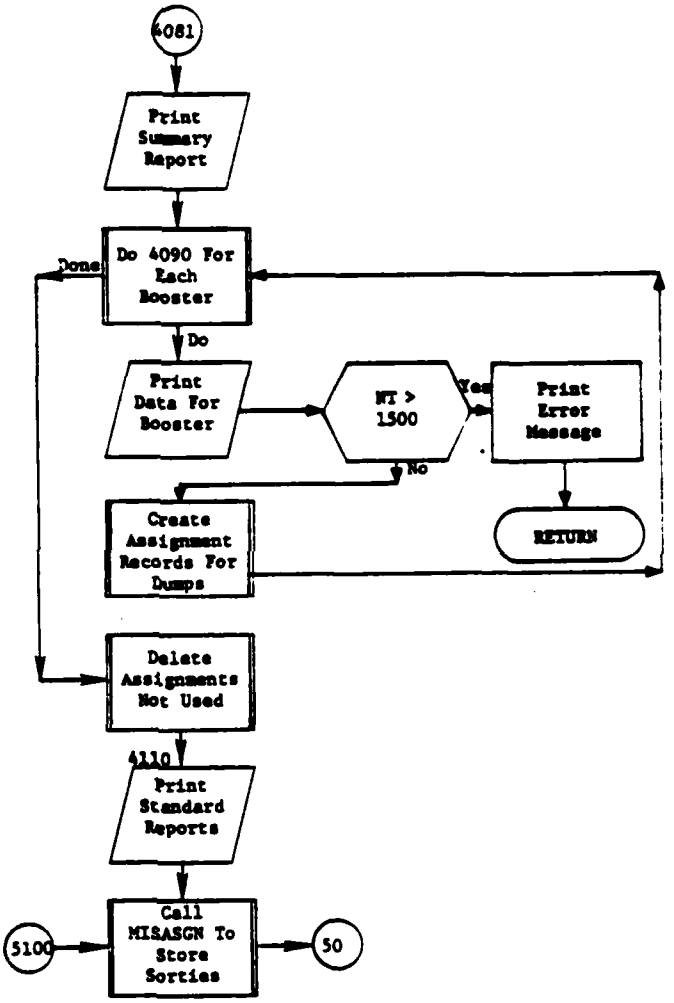


Figure 6. (Part 6 of 6)

2.8 Subroutine TABLINPT*

PURPOSE: This subroutine reads user footprint equation and stores results within the data base.

ENTRY POINTS: TABLINPT

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, C55

SUBROUTINES CALLED: DIRECT, HDFND, HEAD, INSGET, MKAOS, MODFY, NEXTTT, PRAOS, PRINSET, RETRV, STORE, TAOS

CALLED BY: ENTMOD (of FOOTPRNT)

Method:

All user inputs associated with module FOOTPRNT are read by this subroutine. FOOTPRNT recognizes adverbs: ONPRINTS, EQUATE, and REEQUATE (see User's Manual, Volume IV). Adverb ONPRINTS appearance causes the execution of PRINSETS for the controlling of print requests.

Adverb EQUATE defines a complete equation for footprinting use. The names of the equation and the payload are extracted using INSGET. The type of equation is also found.

The start of equation in the EQUATE clause will then be passed to subroutine MKAOS to convert the user's equation into the AOS form. The converted equation, finally, will be stored within the data base as records called "FOOTEQ" and linked to the payload tables matched on the payload name. Any further processors may retrieve the developed equation and execute its contents.

Adverb REEQUATE permits the altering of a portion of an equation developed under a previously executed EQUATE clause. The user specifies what operators and values are to be defined and the subroutine acts accordingly. MKAOS is used to convert the new information to AOS format.

The final equation constructed by these two adverbs is tested by TAOS and printed by PRAOS.

There can also be an IF clause associated with either an EQUATE or REEQUATE clause. The alphabetics in the clause are converted and then the clause is tested to find what values are of RV's and launch azimuth that the equation is restricted to. These values are stored in the FOOTEQ record along with the equation, equation name, and equation type.

Subroutine TABLINPT is illustrated in figure 7.

* Main subroutine in overlay OPTS.

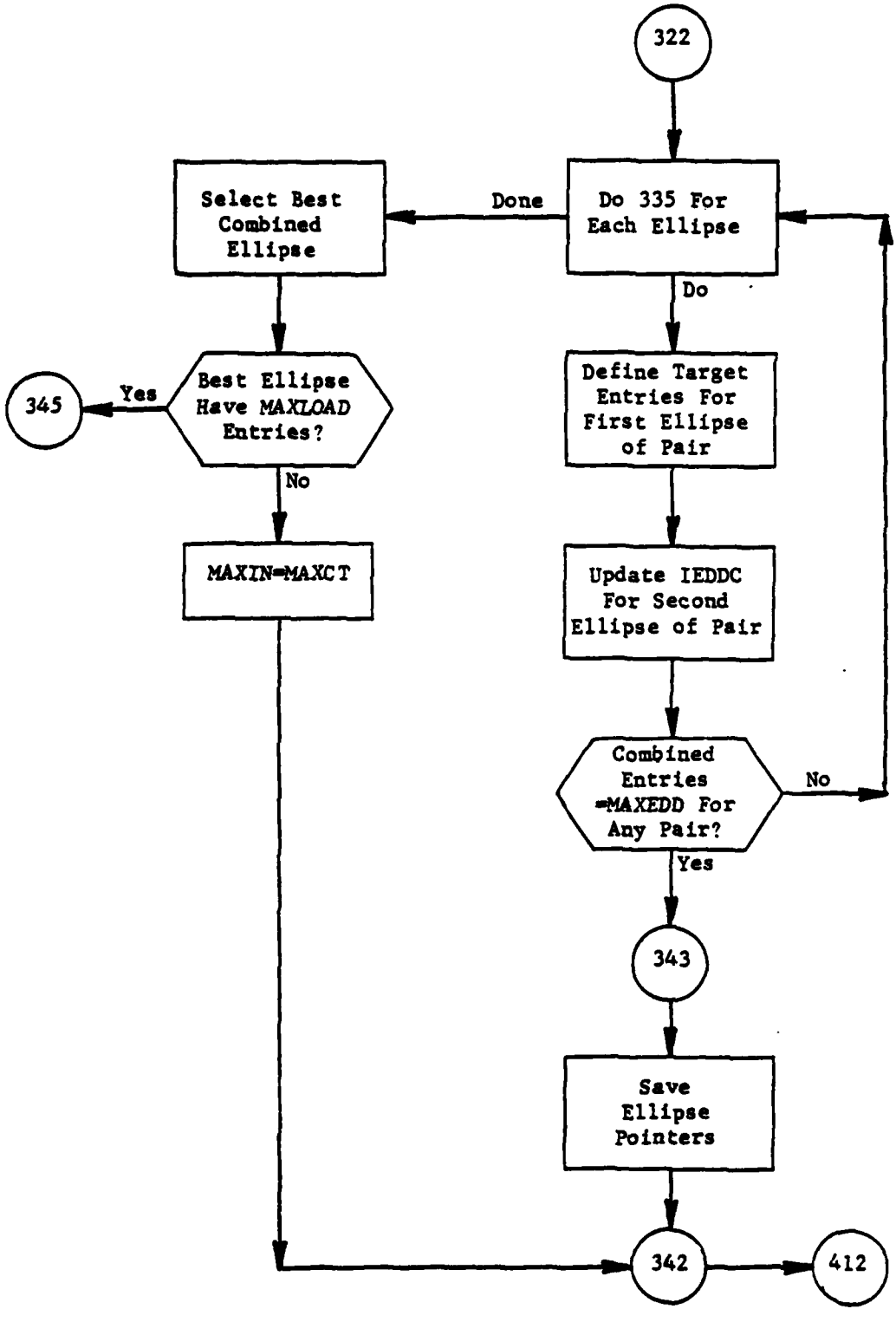


Figure 15. (Part 6 of 8)

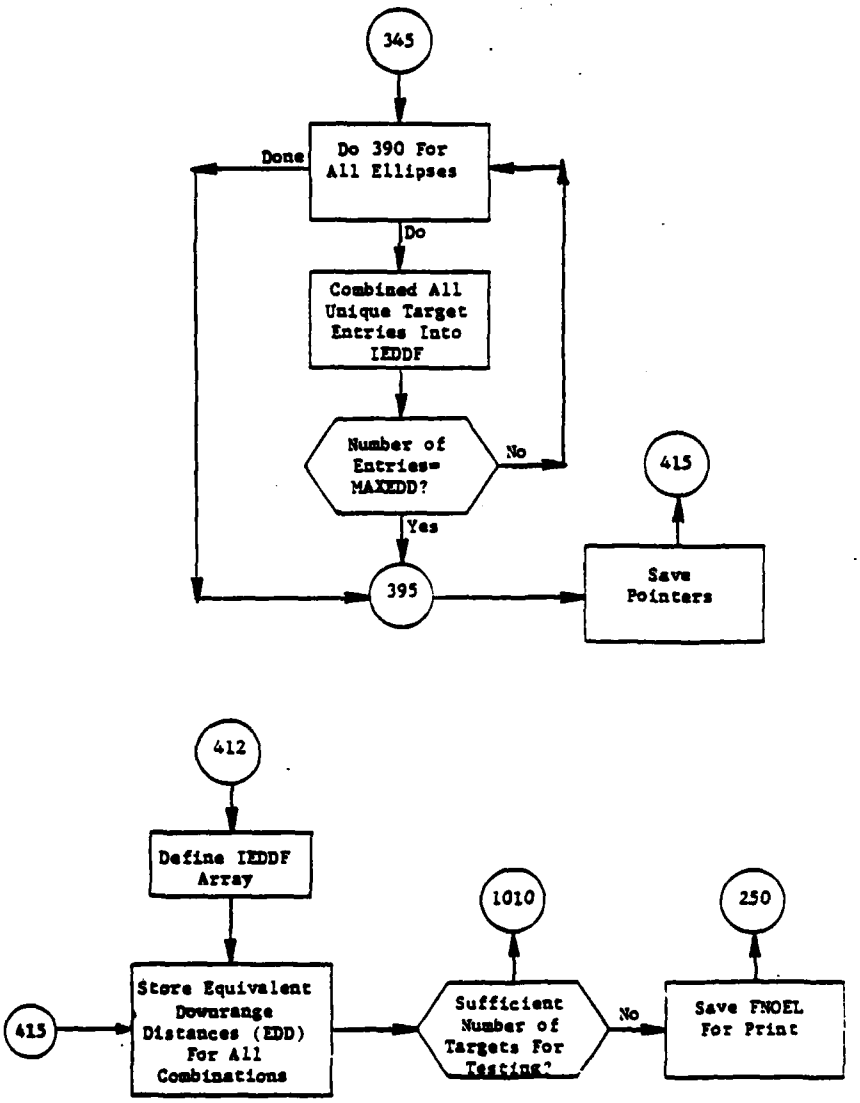


Figure 15. (Part 7 of 8)

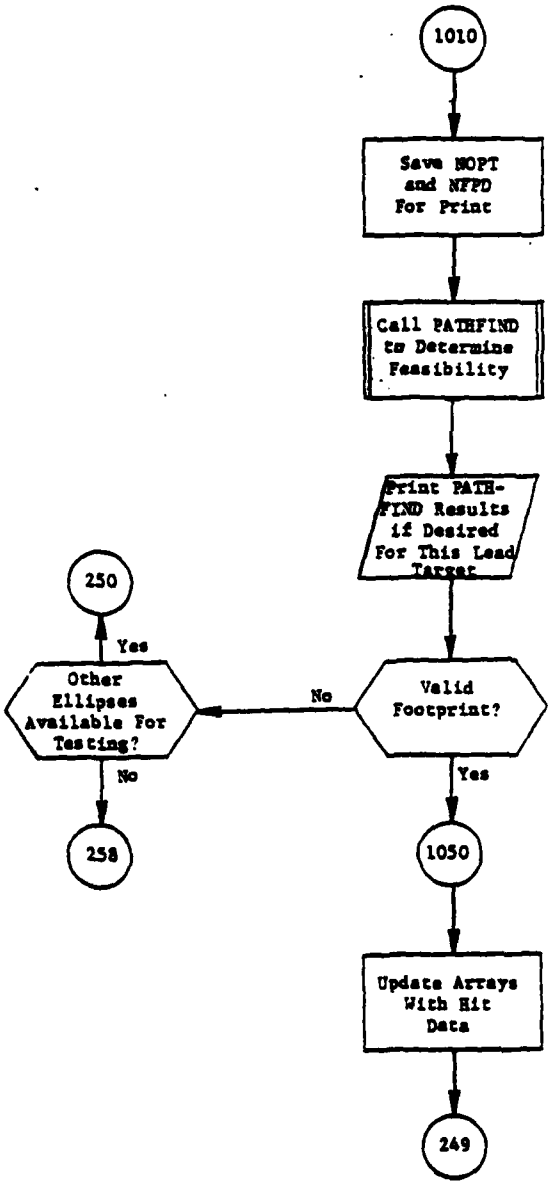


Figure 15. (Part 8 of 8)

2.10.1 Subroutine AXES

PURPOSE: Calculate the semimajor and semiminor axes of five selection ellipses based on calculated fuel load and crossrange/downrange multipliers.

ENTRY POINTS: AXES

FORMAL PARAMETERS: R, AZ distance and azimuth from weapon group centroid to target (n. miles, radians)

COMMON BLOCKS: AXIS, CSAVE, DSQUARE, EARTH, MIRVEQ, SYSMAX

SUBROUTINES CALLED: XAOS

CALLED BY: DRIVER, TABLINPT

Method:

Subroutine DRIVER executes AXES for any target that is being considered as a potential lead target (that is, a target which will be the first assignment from a booster containing MIRV vehicles). Given this assumption, AXES calculates fuel related factors and mathematical contours which when described outlines geographical areas whereby other targets are to be selected as potential elements of a footprint. The contours are range and azimuth dependent and must be calculated anew for each lead target.

Function XAOS computes the downrange/crossrange and downrange/uprange equivalent distance multipliers. The maximum load (ML1) of reentry vehicles to be tossed (number on board at launch minus one) and the sign of the launch azimuth are set.

Based on the applicable system, the maximum booster range is calculated and compared against the range to the lead target. If the range to the lead target is greater than maximum, fuel is borrowed from the bus for range extension and the onboard fuel load (parameter FUELOAD) decremented accordingly and fuel consumption rates calculated.

Potential footprint contours are best approximated by an ellipse. Therefore, given a lead target the definition of ellipse(s) serves as an appropriate screening mechanism for selecting potential targets for footprint assignment. AXES, then, calculates semimajor (SMA) and semiminor (SMI) axes for five separate ellipses using assumptions based on the extremeties of deployment. Figure 16 presents the configuration for ellipses labelled 1, 3, and 5. Ellipse 2 equals ellipse 1 with a clockwise rotation of 45°; ellipse 4 equals ellipse 5 with a counterclockwise rotation of 45°. Targets to be considered for assignment

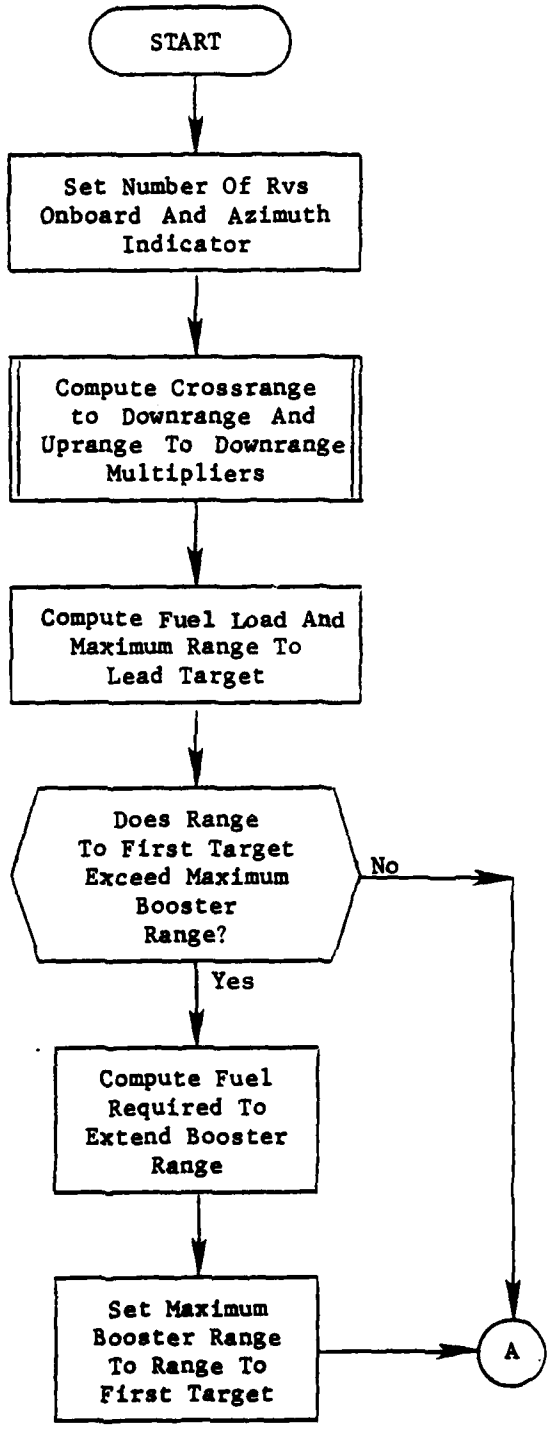


Figure 17. Subroutine AXES (Part 1 of 2)

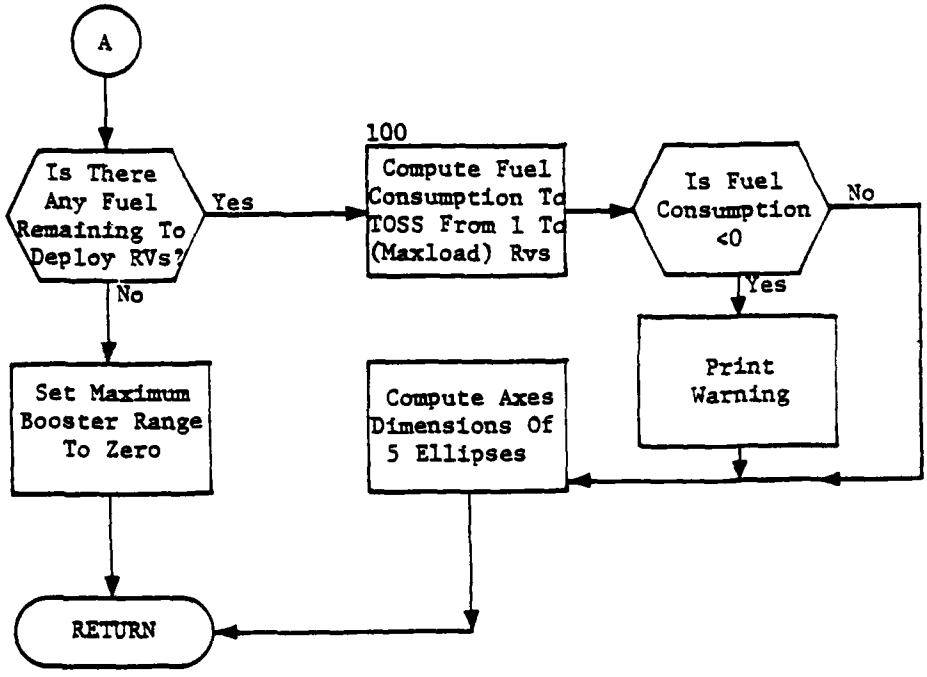


Figure 17. (Part 2 of 2)

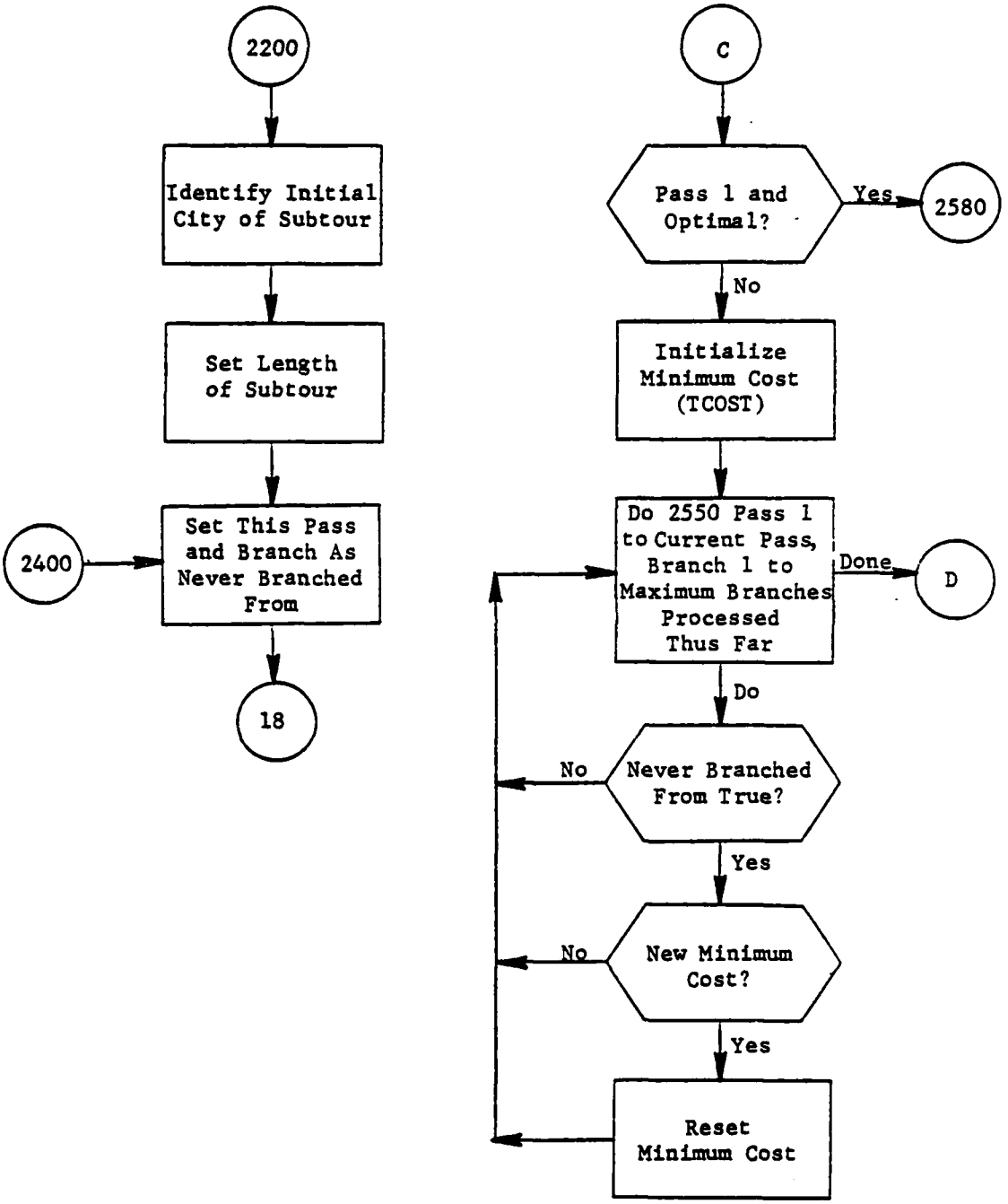


Figure 23. (Part 5 of 8)

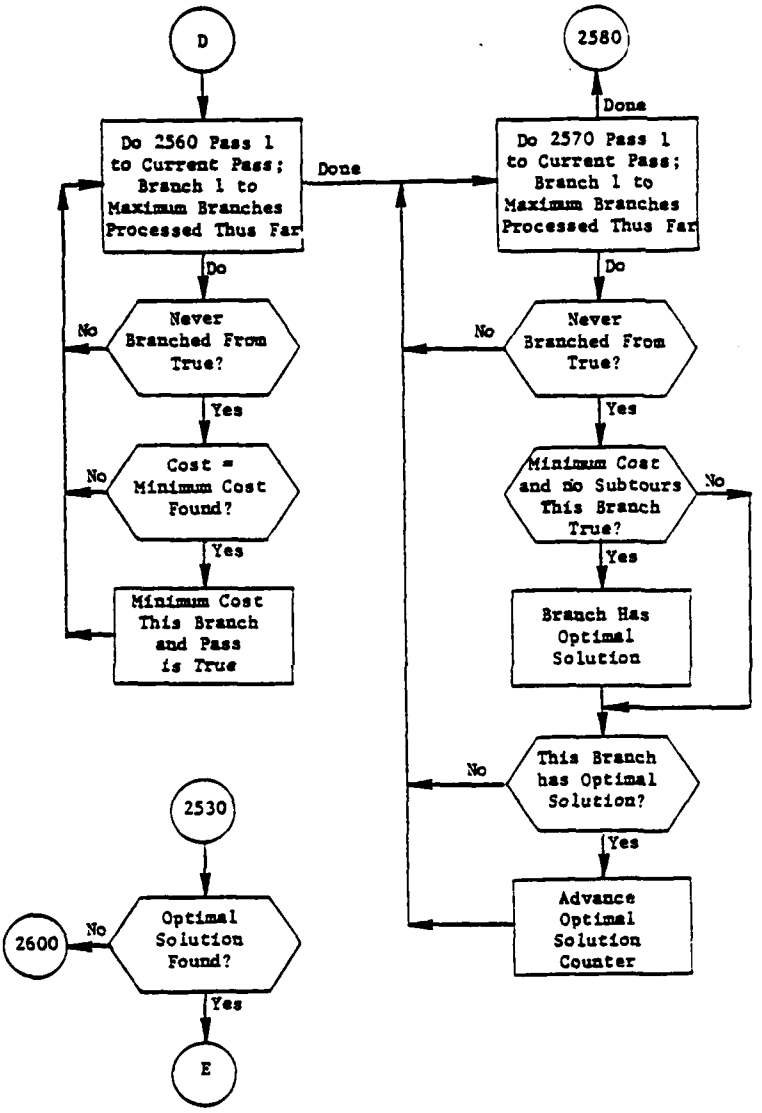


Figure 23. (Part 6 of 8)

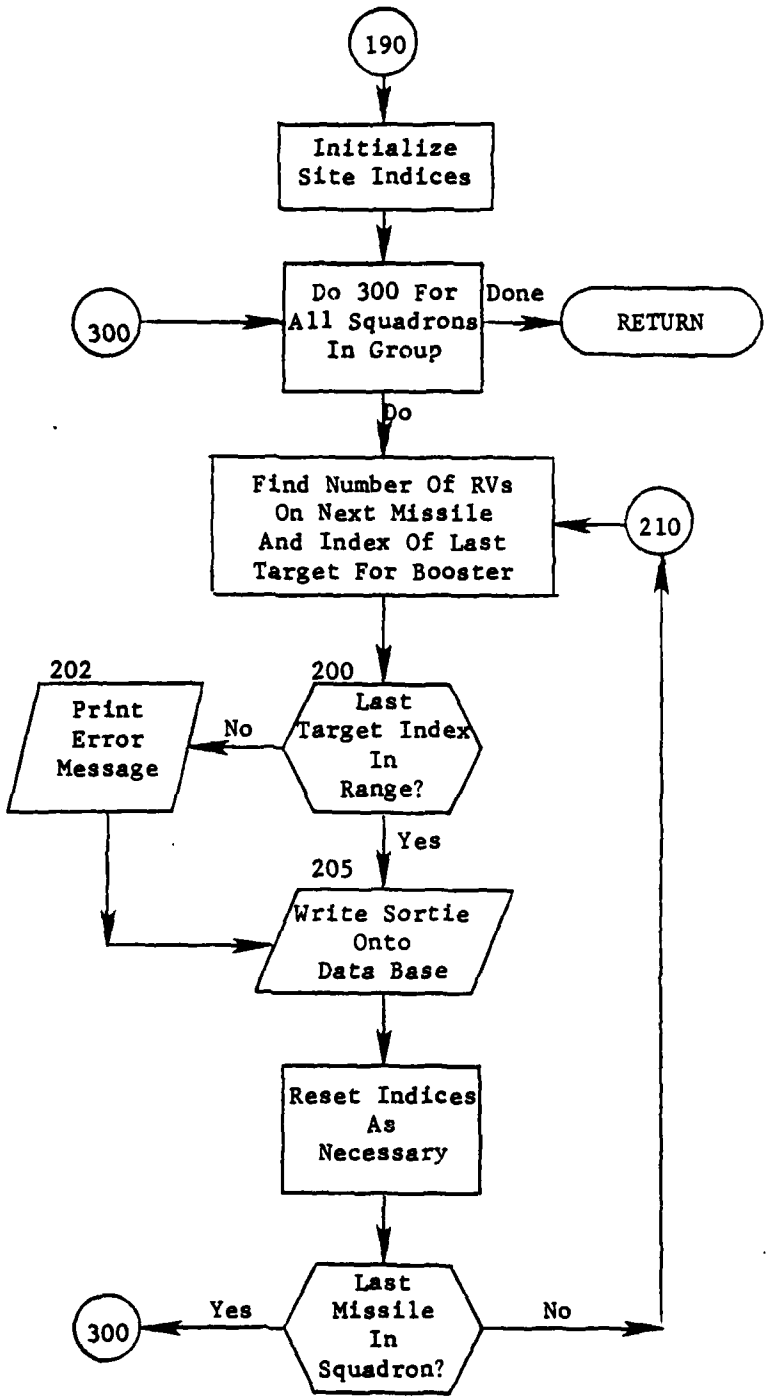


Figure 25. (Part 3 of 3)

SECTION I1. MIRVDUMP MODULE

I1.1 General Purpose

MIRVed missile sorties which have more than one weapon impacting at the same point (dumping) are processed to determine alternate impact points. Three methods are used to select these new impact points. The first method examines the complexed and single point targets in the QUICK target base (max 4,000 targets) to determine if a subset of these targets can be found which is capable of being footprinted. This subset always contains the original targets chained to the sortie. If no suitable footprint is found using this set of targets, the installations chained to the dumped target's complex are checked and, if a non-collocated installation is found, the weapon is placed on that installation. If there are not sufficient installations within the dumped target's complex (e.g., dumping occurred on a single point target) the other complexes attacked by this sortie are searched for installations which do not have a weapon impacting on them. Since weapons are assigned to these complexes in the original footprint, no checks are made to determine if the resultant impact points can be footprinted. PLANOUT change cards (CCARDs) are output for the new impact points to modify the data base in a subsequent PLANOUT run.

I1.2 Input

MIRVDUMP may be executed after processing by FOOTPRNT has been completed. $ SET cards are used to set control bits 20 thru 25 for internal control of parameters. If bit 20 is set, the length and width of the initial area searched for targets to footprint will be increased by a factor of 1.5. Bits 21 thru 25 correspond to a counter limit of 250, 500, 750, 1,000, and 5,000. This counter restricts the number of trials made to find a suitable footprint.

The number of targets considered by MIRVDUMP is restricted to 4,000. Subroutine TGTLIM in the UTILITY library has a common block, TGTLM, which contains the number of sets of DESIGs input and the pairs of DESIGs identifying targets to be omitted from consideration during MIRVDUMP processing. The maximum number of sets that can be input is 30. The data set AB, AC, DE, EK, CC, CC will omit all targets whose DESIGs start with AB through AC, DE through EK and CC. The DESIGs to te omitted must be selected by the user and input by the maintenance programmer. If no inputs are made the first 4,000 targets will be used.

I1.3 Output

The outputs of the MIRVDUMP module consist of a standard report of footprints which have been processed, a list of change cards and a file containing the change cards. The standard report is output on file code 12. All user and programmer error messages will also be on this file. The list of change cards will be printed on file code 13. File code 17 will also contain the change cards. This file can be used as an input file to PLANOUT.

I1.4 Concept of Operation

Each weapon group is checked to determine if it is a MIRVed missile group. If it is, each sortie is checked to determine if more than one weapon on the EVENT chain for a given SRTYTB has the same impact point. If multiple impacts at the same location are found and the weapon locations are not fixed by PREPALOC FIX clauses, the dumped footprint processing will be accomplished.

I1.4.1 The Target List Processing. Prior to processing any dumped footprints, the QUICK target base (TARCDE records) is read into internal storage (TGTDAT array). The reduction of the target base to 4,000 is accomplished by *DESIGs*. *Since DESIGs are data dependent, a change in* the data base may require a change in the TGTLIM subroutine. Selection of the targets is accomplished by subroutine GETGT.

I1.4.2 Candidate Target Selection. The targets which are reasonable candidates for footprinting are selected from the TGTDAT array for each dumped footprint using the MIRV weapon system capabilities and the location of the lead target in the sortie. These targets are stored in the RAIDAT array. These candidate targets are then sorted into geographic areas determined by distance from the lead target and the targets in each area are further sorted into two sets: one which is not targeted and one which has weapons assigned. The objective of this sorting process is to increase the probability of obtaining a set of targets which can be footprinted with the minimum number of trials. Subroutine NEWCORD selects the candidate targets.

I1.4.3 Footprint Verification. The original targets and sufficient targets from the RAIDAT array to provide one target for each weapon in the sortie are combined as a possible footprint. Each combination of weapons is checked by subroutine FOOTCHCK to determine if that set of targets can be footprinted. This process of checking for footprints continues until one of the following conditions occur: (1) three footprints are found, (2) all target combinations have been tried or (3) the number of trials exceeds the allowable number of trials (MCOUNT which can be changed by bits 21-25 of a $ SET card).

I1.4.4 Alternate Impact Point Determination. If no footptints are found using the procedure in the previous paragraph, each complexed target in the original footprint is searched for installations which are not collocated with the dumped installation. If the complex has more than one weapon assigned to it, one of the untargeted installations in the complex is designated as an impact point for the dumped target. If this procedure does not provide sufficient impact points, other installations in the dumped complexes and installations in the other complexes attacked by the sortie are used as impact points for the dumped weapons. If no suitable location for the weapons can be found, a message is output and processing for that sortie is complete.

I1.4.5 Output Processing. If a footprint was found using the targets in the TGTDAT array, the footprint found with the most unassigned targets is used to prepare change cards (CCARDS). If equal numbers of unassigned targets are in more than one footprint, the footprint with the highest value of unassigned targets is used. If a footprint was not found using the lead targets, the installations within the complexes are used as impact points as explained in the previous paragraph. The resultant change cards are output to file code 13 for printing and to file code 17 for use as an input to PLANOUT.

I1.5 Identification of Subroutine Functions

I1.5.1 Subroutine ENTMOD (MIRVDUMP). The MIRVDUMP subroutine controls the processing of the MIRVDUMP module. It calls GETGT to read in the target list, and it walks the WEPGRP, MYSRTY, and EVENT chains to find the dumped footprints. Then it calls NEWCORD to select the candidate targets for the dumped sortie. COMBO is called to select a combination of targets for potential footprinting and DRIVER is called to initiate the footprint checking process. If no footprints are found, the MIRVDUMP module accomplishes the processing required to determine alternate impact points within the original footprint complexed targets. This module also outputs the footprint processing report and writes the change card report and file.

I1.5.2 Subroutine GETGT. The QUICK target list (LISTXX Chain) is read into internal storage prior to processing any footprints by the subroutine GETGT. The DESIGs and coordinates of all lead elements of complexes and single point targets which are not excluded by user inputs are stored in the TGTDAT array.

I1.5.3 Subroutine NEWCORD. Subroutine NEWCORD fills the RAIDAT array with targets to be checked for each dumped footprint. The lead target for the footprint and the other targets in the footprint for the sortie are placed in the array on the first call to NEWCORD. During the second

call to NEWCORD, the targets in the TGTDAT array which are likely candidates for completing the footprint are transferred to the RAIDAT array. When this information has been completed, the targets are sorted and rearranged to allow the most likely candidates for footprinting to be tested first.

I1.5.4 Subroutine FOOTCHCK. Subroutine FOOTCHCK is used to check if a set of input targets can be hit by the weapons from a MIRV booster. Using available energy on the MIRV bus and energy requirements as input to the FOOTPRNT module (FOOTEQ records) all combinations of targeting sequences of the targets are checked to determine if a feasible footprint can be found. This module performs a function similar to the PATHFIND module in FOOTPRNT. The primary difference is that in PATHFIND, the lead element of the footprint must always be targeted with the first weapon released from the bus.

I1.6 Common Block Definition

Common blocks used by MIRVDUMP are given in table 1.1. Common blocks C10, C15, C30, are described in Program Maintenance Manual, Vol I. Common blocks C55, MIRVEQ, CSAVE, AXIS, EARTH, SYSMAX, and DSQUAR are associated with subroutines in FOOTPRINT and are described in the description of the FOOTPRNT module.

Table 1.1. *Module MIRVDUMP Common Blocks*
(Part 1 of 2)

BLOCK	Variable or Array	Description
EDD	EDD (25,25)	The target cost matrix passed to FOOTCHCK
	MAXIN	The number of RVs on a single booster
	RFIRST	Range of first target from launched point
GRPDAT	IGPREF (250)	IDS reference codes for each group on the WEPGRP chain.
	MFLAG (7)	Flag which indicates MIRVed or non-MIRVed system. One bit is used for each group.
HIT	HIT (16)	An index of the targets in the footprint which is used to index the EDD array.
MISNR	KSEQ (16)	Index numbers to the INDEX array for targets selected for the footprint
MIST	IINDEX (16)	Index used to designate targets which are ordered by range.
RAIDAT	INDEX (1000)	DESIG for candidate target
	R (1000)	Range from launch point to candidate target
	THETA (1000)	Launch angle to candidate target
	NTG (1000)	Search region to target in relation to lead target of footprint
	NT	Number of candidate targets
	TGTLAT(20)	Latitude of target in initial footprint
	TGTLONG (20)	Longitude of target in initial footprint
	WLAT	Latitude of launch position
	WLONG	Longitude of launch position
	NTI (11)	Indexes for starting locations in array of geographically sorted sets of targets
TGTDAT	ITREF (4000)	DESIG of QUICK target
	TGLAT (4000)	Latitude of QUICK target
	TGLON (4000)	Longitude of QUICK target
	FACTOR	Scale factor for determining area to be searched for candidate targets
	NTF	Index for next storage location of an assigned target (starts at beginning of TGTDAT arrays)

Table 1.1. (Part 2 of 2)

BLOCK	Variable of Array	Description
	NTL	Index for next storage location of unassigned target (starts at end of TGTDAT arrays and runs backwards)
TGTLM	ITLIM	No. of sets of DESIGs to be excluded
	DESLIM (2,30)	Pairs of DESIGs which define targets to be eliminated from the QUICK target list to reduce the size of the target list in MIRVDUMP to 4000.

I1.7 Subroutine ENTMOD

Purpose:	This subroutine controls the processing of the MIRVDUMP module, checks for dumped footprints and processes targets within complexes to select required impact points.
Entry Points:	ENTMOD (First subroutine executed in overlay link MDUMP)
Formal Parameters:	None
Common Blocks:	C10, C15, C30, EARTH, GRPDAT, HIT, MISNR, RAIDAT, SYSMAX, TGTDAT
Subroutines Called:	AXES, COMBO, DIRECT, GETGT, HEAD, HDFND, NEWCOR, NEXTTT, PTIME, RETRV, SETDAT, SLOG, SWTCHT
Called By:	COP

Method: Initialization and Target Selection

Initialization and reading of user inputs to determine the number of checks desired and the area to be searched are accomplished first in the ENTMOD module. The weapon group (WEPGRP) chain is then read and the group is checked to see if it is a MIRVed system and if it has some weapons which are not fixed. Switches are set for the MIRVed systems and the direct reference code for the group in the data base is stored; a call is then made to GETGT where the targets to be used for footprinting are selected and stored in TGTDAT common block.

Each MIRVed group is sequentially processed through the following events.

Checking for Dumped Sorties

The sortie event chain (EVENT) is read for each sortie (MYSRTY) and each impact event for the sortie is checked to determine if any of the impact points are the same. As the checking progresses, the distinct impact points are moved into the RAIDAT common block. When duplicate impact points are found, the location of that target in the RAIDAT common block is stored in IDUMSV array and the sortie event number is stored in the ISEQSV array. The number of dumped targets (IMDUMP) is also increased by one. After all events have been processed subroutine NEWCORD is called, if a dumped sortie was found, to compute range and azimuth information for the sortie events stored in the RAIDAT common block. IMDUMP is then redefined to equal the number of RVs on the missile booster.

Selection of Candidate Targets and Footprint Testing

Footprint data for the sortie is obtained using the SETDATA and AXES subroutines. NEWCORD is then called a second time to select the targets to be checked for footprinting from a geographic area determined by the impact point of the first target in the footprint and the weapon system footprint characters of the booster missile. These targets along with the required range and azimuth data computed in NEWCORD are also stored in the RAIDAT common block.

The initial targets in the footprint and as many other targets from the RAIDAT common block as are required to complete the footprint (INEED) are selected for a trial footprint. The subroutine COMBO selects a different combination of targets for each trial and sets a switch when all combinations have been tried. Since NEWCORD sorted the candidate targets into subsets depending on their distance from the initial target and their assigned (A) or unassigned (U) status, twenty different subsets of targets must be checked. Each new subset is added to previously checked targets and all new combinations of the resultant set of targets are checked. This process is controlled by setting the values for ISTART, ISTOP, and IEND. These variables refer to locations in RAIDAT common block. Each combination of targets is passed to the DRIVER subroutine which calculates data for the energy requirements matrix. This data is then used by subroutine FOOTCHCK to determine if the selected targets can be footprinted. After three footprints have been found or all combinations have been tested (whichever comes first) the process is terminated.

Impact Point Selection

After each feasible footprint is found, the value of the unassigned targets (targets which have not previously been hit by any weapon) in the footprint is determined and the DESIGs of the footprint with the highest unassigned value are stored in ISAVE array. If no feasible footprints were found, alternate impact points for the dumped weapons are searched for from within the complexes which contain more than one installation. The dumped target complex is searched for an installation which is not located at the impact point. A dumped weapon is then assigned to this installation (location in RAIDAT is stored in ISAVE array). Other non-collocated installations are saved in the potential impact array (location in RAIDAT is stored in the KSAVE array). Potential impact points are saved until the sum of the number of impact points stored in ISAVE array and the number of impact points stored in the potential impact array (KSAVE) equals the required number. All dumped complexes are processed and other complexes in the original footprint are processed if additional potential impact points are needed. The dumped weapons remaining after attempting to reassign one dumped weapon to each dumped complex are assigned impact points in the order in which the potential impact points were found. This is accomplished by moving the locations in the RAIDAT common block from the KSAVE array to the ISAVE array. If not enough impact points are located to meet the requirements, a message is printed stating that no footprint was found.

Preparation of CCARDS

The sortie number, event sequence number, and new DESIG are provided, with leading zeros where required and encoded into the CCARD format using the subscript information contained in the ISAVE array. The variable ISEQSV(I) contains the event sequence number.

When all MIRVed groups and their associated sorties have been checked, the execution of the module is complete.

Figure 25.1 is a flow diagram of the ENTMOD module.

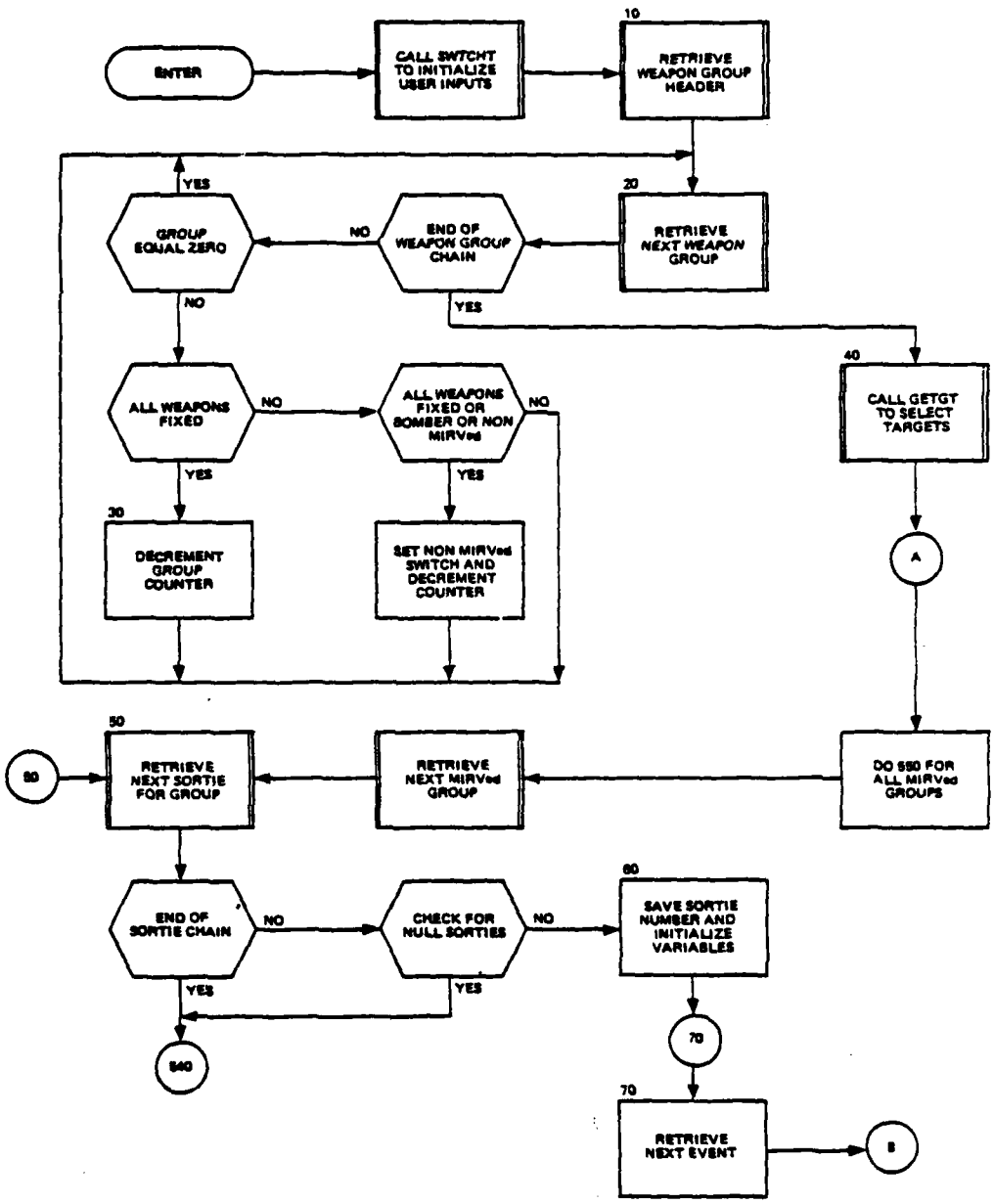


Figure 25.1. Subroutine ENTMOD (Part 1 of 6)

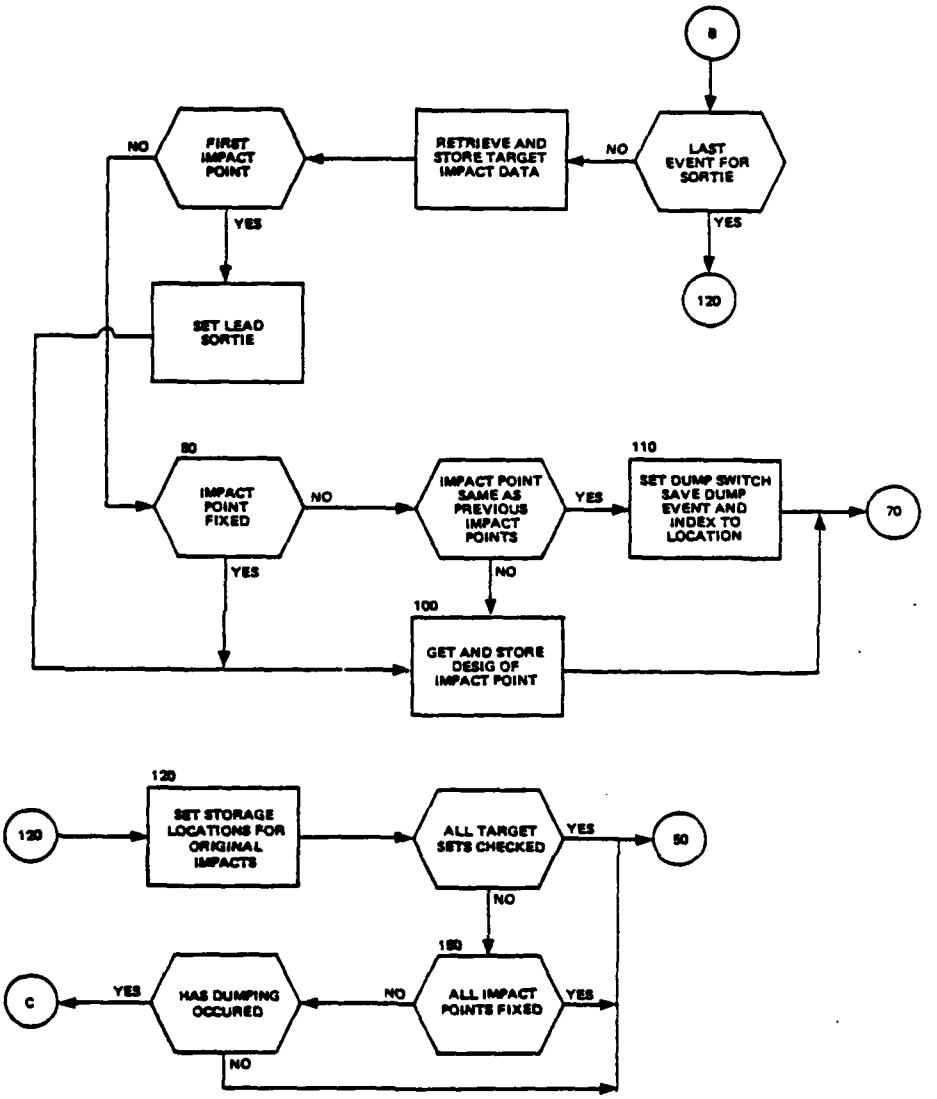


Figure 25.1. (Part 2 of 6)

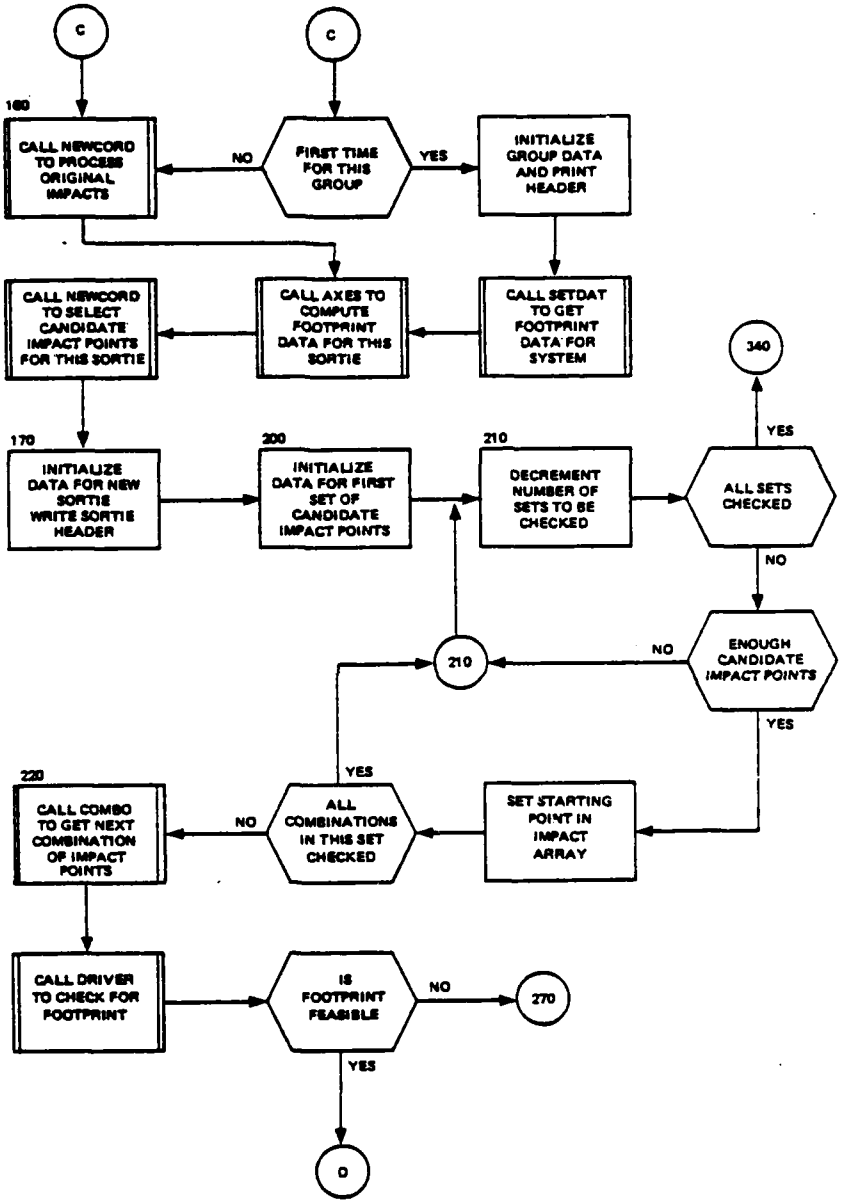


Figure 25.1. (Part 3 of 6)

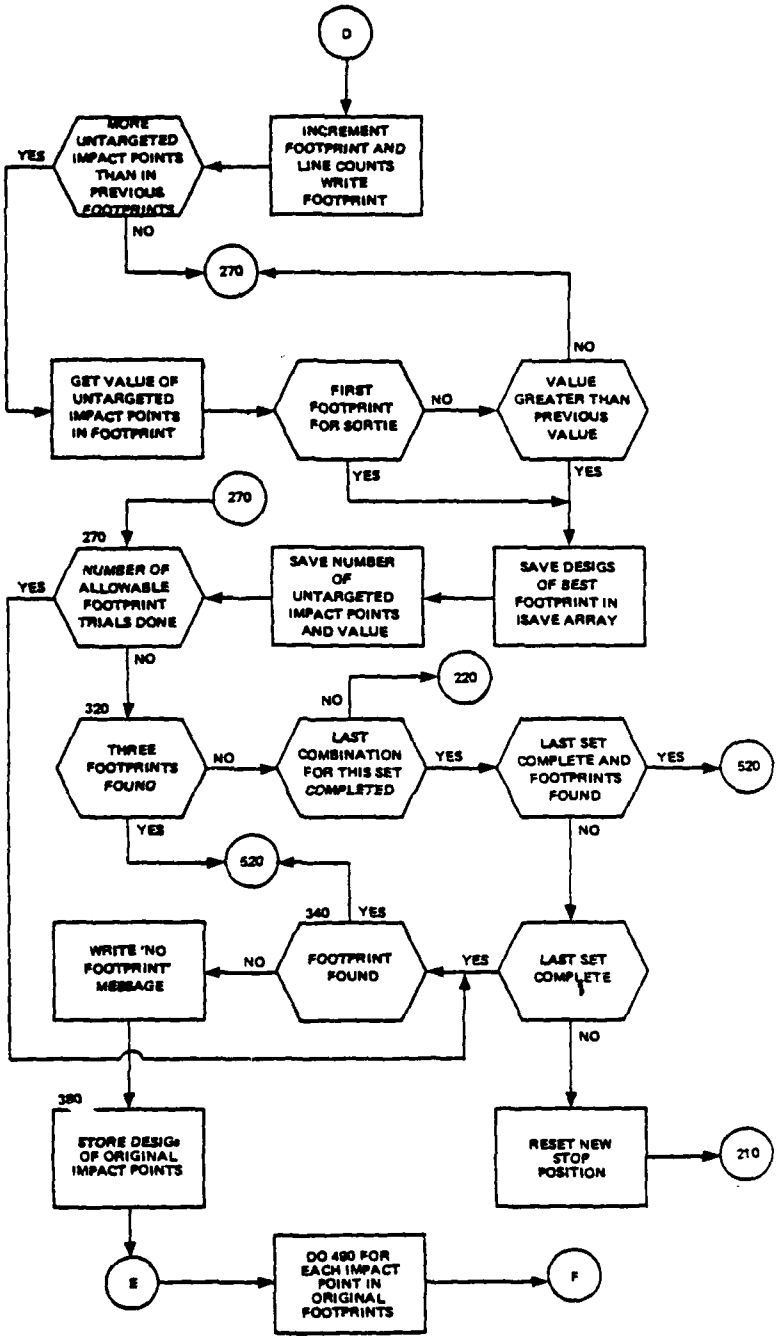


Figure 25.1. (Part 4 of 6)

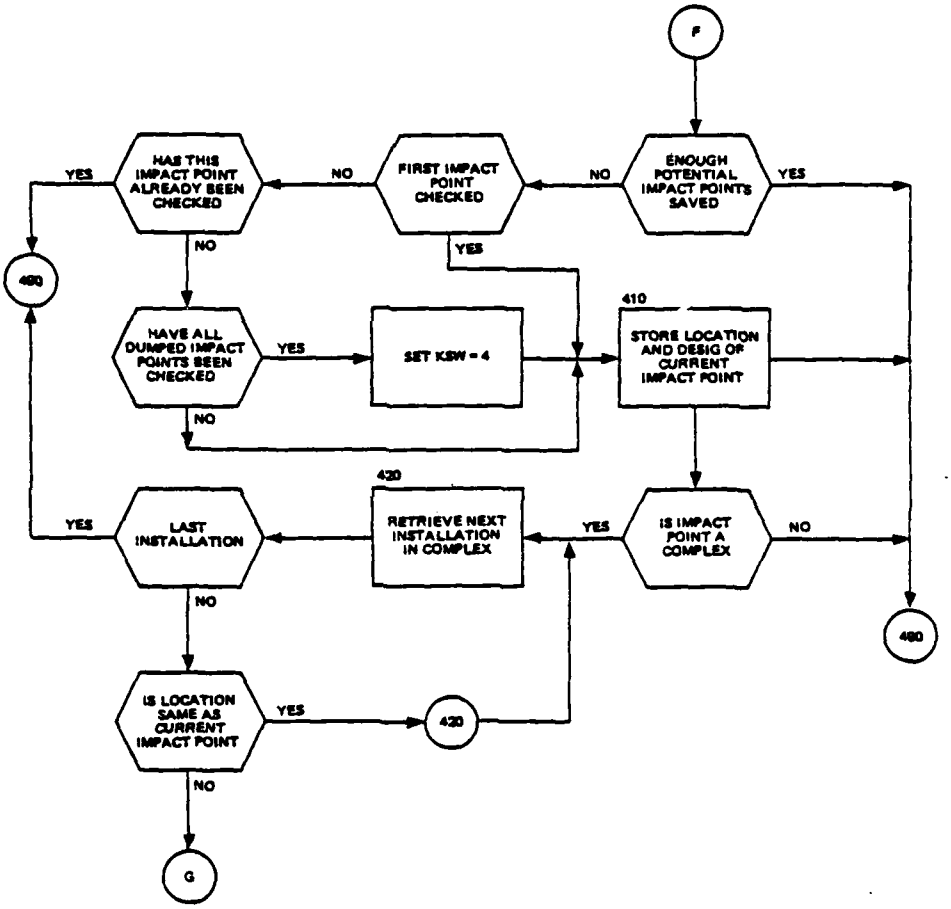


Figure 25.1. (Part 5 of 6)

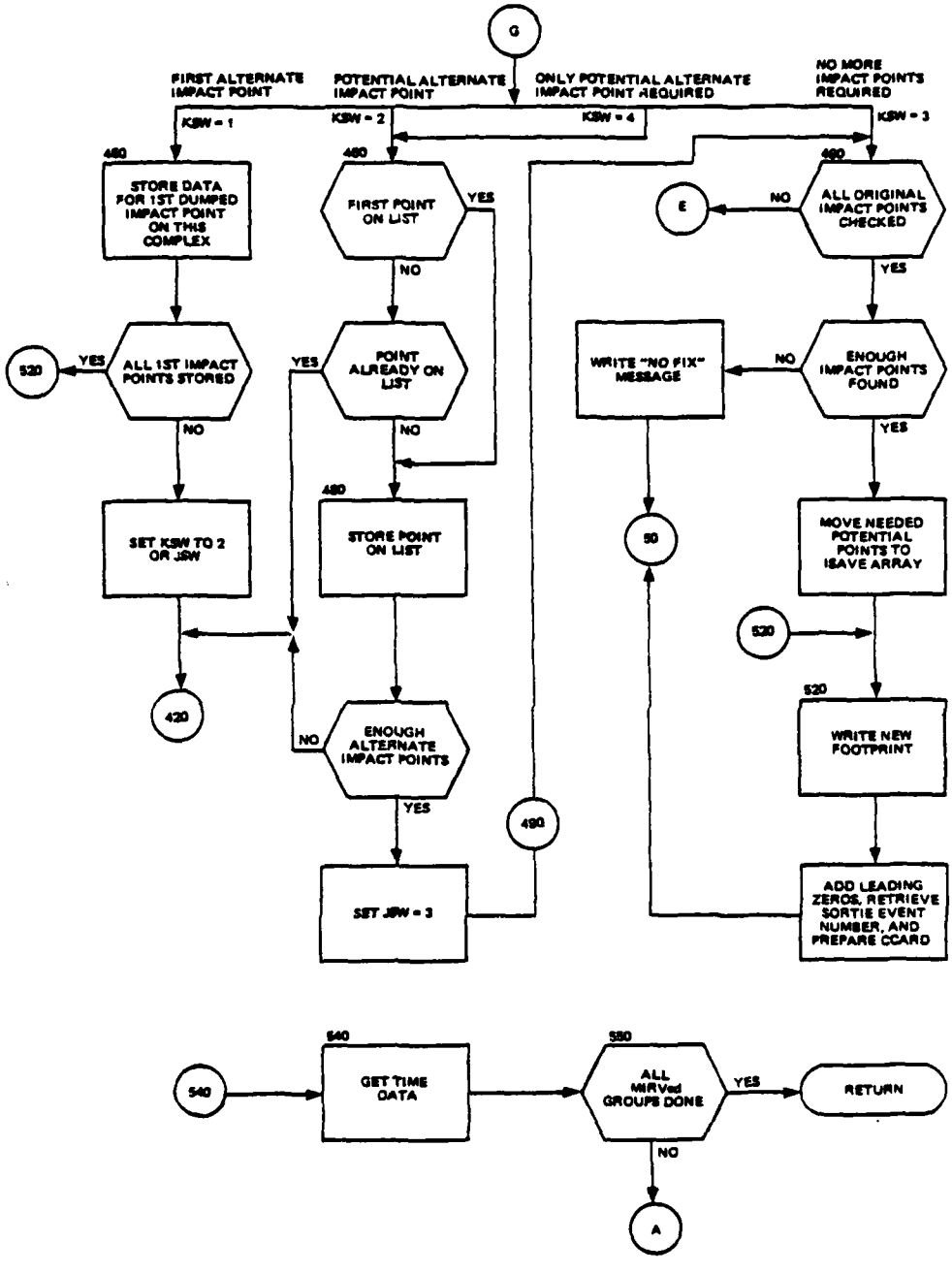


Figure 25.1. (Part 6 of 6)

I1.8 Subroutine GETGT

PURPOSE: The GETGT subroutine reads targets from the QUICK data base, and eliminates those specified by the user in subroutine TGTLIM to reduce the size of the data base. The selected targets are checked to determine if they are programmed for any weapons and an A or U suffix is added to the stored DESIG to denote this status. If over 4,000 targets are eligible, the first 4,000 targets will be used.

Entry Points: GETGT

Formal Parameters: None

Common Blocks: C10, C15, C30, TGTDAT, GRPDAT, EARTH, TGTLM

Subroutines Called: HDFND, RETRV, NEXTTT, DIRECT, TGTLIM

Called By: MIRVDUMP (ENTMOD)

Method:

During the initialization, GETGT calls subroutine TGTLIM to initialize the user-supplied data for eliminating targets. This data is stored in common block TGTLM. Each target on the LISTXX target chain is processed to determine if it is an eligible target and if it has weapons already assigned to it. One condition for an unassigned target is that no group was assigned. If a group was assigned, checks are made to determine if a sortie is assigned. If no sortie is assigned, the target is unassigned. In either case, the unassigned target is assigned a U suffix on the DESIG and is stored in the last available space in the TGTDAT common block. If the target is assigned a sortie, an A suffix is added to the DESIG and the target is stored in the first available position in the TGTDAT common block. The targets are stored from both ends of the array toward the middle. The checking and storing process continues until all targets have been checked or until the two sets of targets meet at the middle of the array. When the array is full a message is output stating that there are over 4,000 targets available and target selection stops. If the target list is exhausted, the unassigned targets are moved up in the array to form one continuous target array. A return is made to the MIRVDUMP module and the targets in the TGTDAT common blocks are used for processing the dumped footprints.

Figure 25.2 is a flow diagram for the GETGT subroutine.

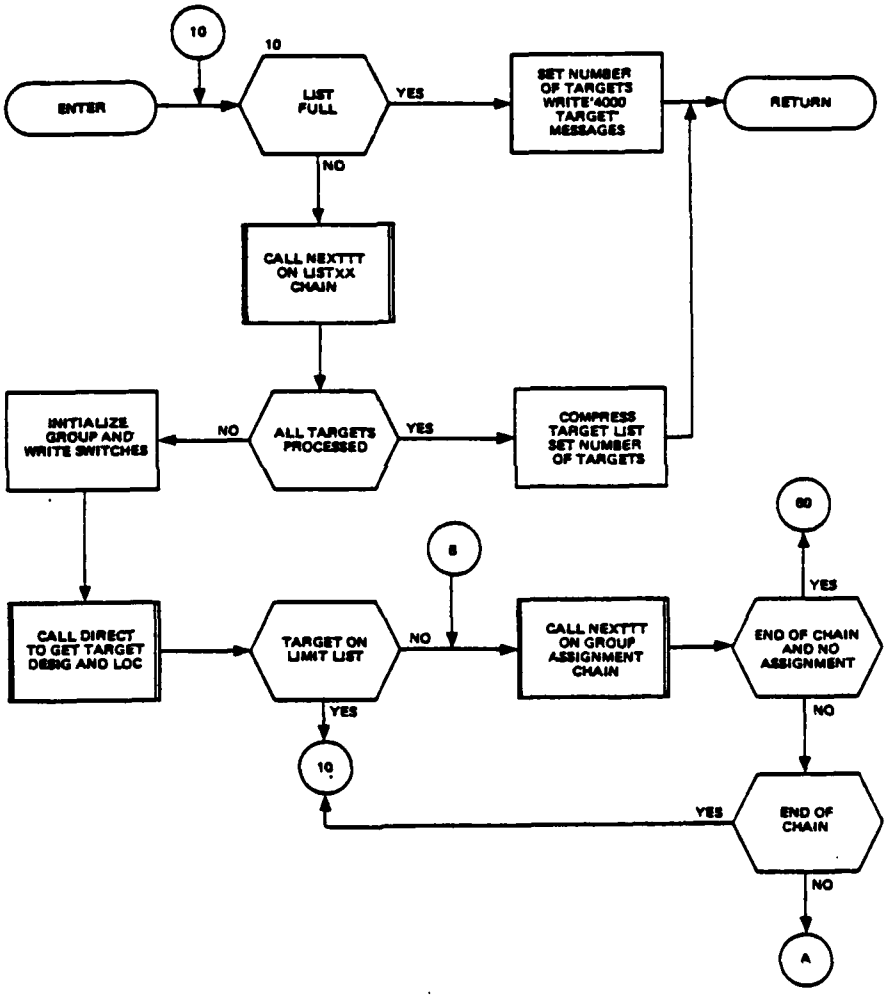


Figure 25.2. Subroutine GETGT (Part 1 of 2)

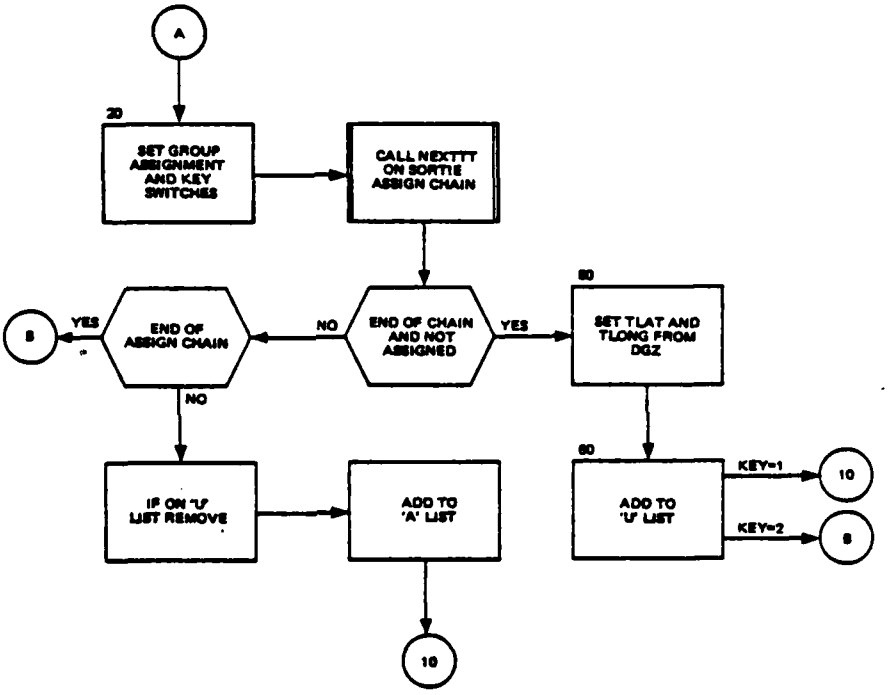


Figure 25.2. (Part 2 of 2)

I1.9 Subroutine NEWCORD

PURPOSE: The NEWCORD subroutine selects candidate targets for the footprint under consideration. It calculates range and azimuth data for these targets and sorts them into subsets according to their distance from the first target in the original footprint.

Entry Points:	NEWCORD
Format Parameters:	IG - The number of distinct impact points in the dumped footprint
Common Blocks:	C10, C15, C30, AXIS, RAIDAT, EARTH, TGTDAT
Subroutines Called:	ORDER, REORDER
Called By:	MIRVDUMP

Method:

On the initial call, the NEWCORD subroutine calculates range and azimuth data for the original impact points in the footprint and stores them in the RAIDAT common block. On the second call, maximum uprange, downrange and crossrange distances for the footprint are calculated. One fifth the value of these ranges are used to sort the selected targets into subsets which are increasingly distant from the first impact point in the original footprint. This subsetting is used to increase the likelihood of selecting a feasible footprint with the early choices of target combinations. All targets outside the maximum uprange, downrange, and crossrange distances are eliminated from consideration for the footprint being processed. The range azimuth and subset data for the selected targets are calculated and the targets are stored in the RAIDAT common block. When all targets have been processed, they are sorted by the subsets in which they are located, separated into unassigned and assigned groups within each subset and are reordered according to these subsets. The endpoints of each subset are stored in the NTI array. The targets with the highest subset number are closest to the first target of the original footprint. A maximum of 1,000 targets can be considered for each footprint.

The flow diagram for the NEWCORD subroutine is shown in figure 25.3.

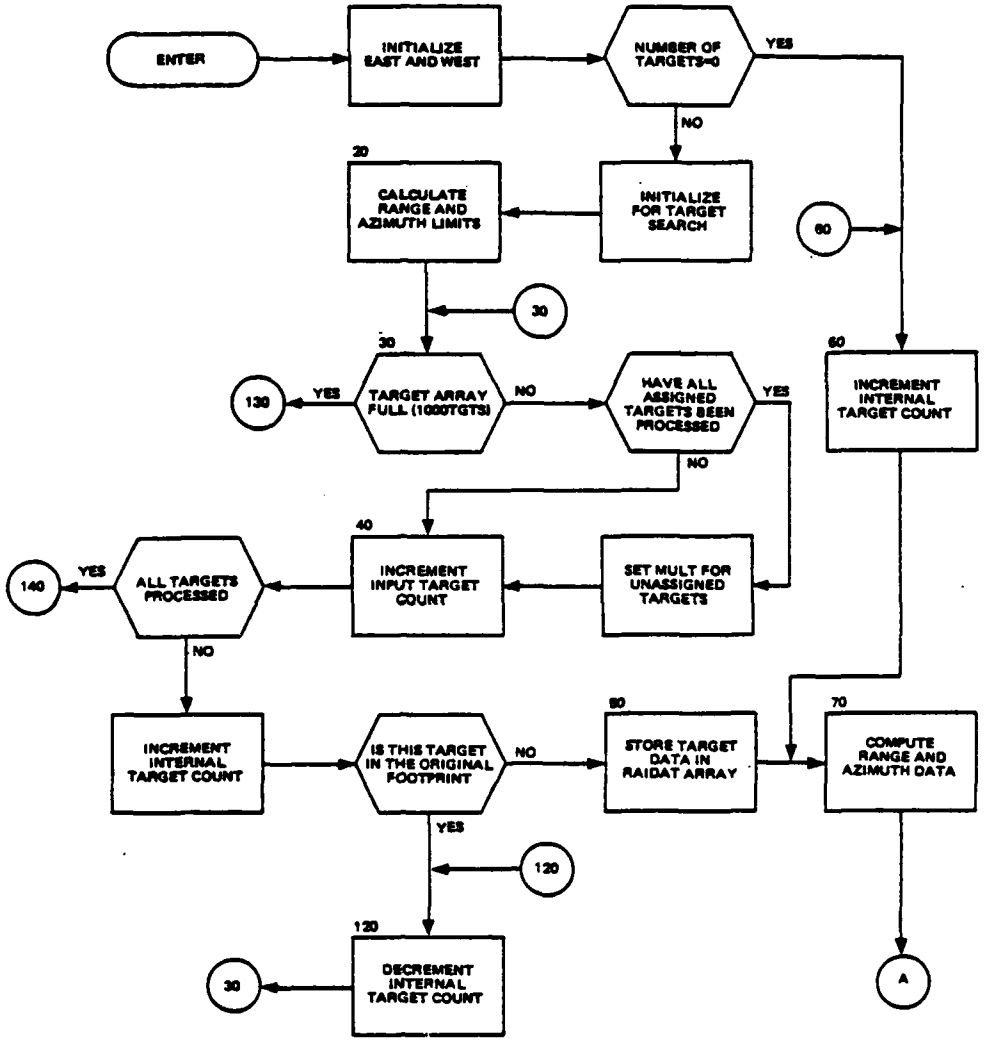


Figure 25.3. Subroutine NEWCORD (Part 1 of 2)

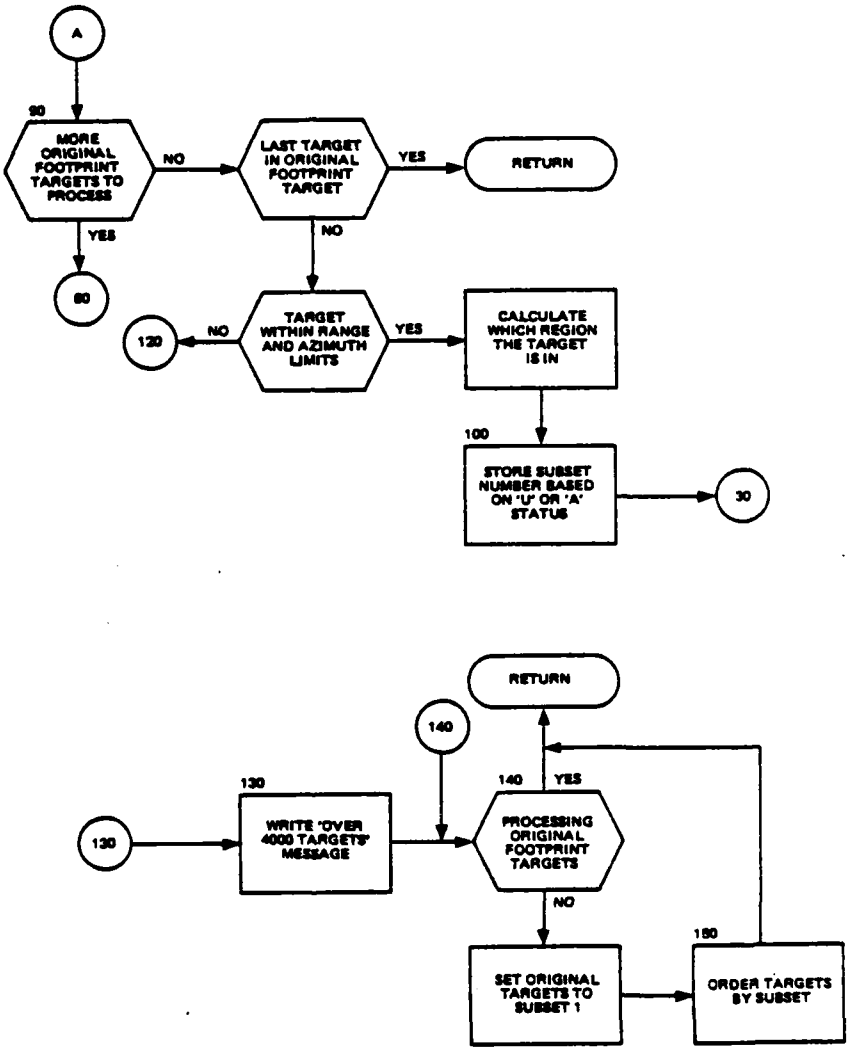


Figure 25.3. (Part 2 of 2)

I1.10 Subroutine COMBO

PURPOSE: COMBO selects all combinations of n targets taken m at a time. The value of n and m are determined by the calling variables. If some combinations have been previously selected, they can be excluded from consideration by use of the ISTOP input variable.

Entry Points: COMBO

Format Parameters:

- MIRVTGT - Number of distinct impact points in original footprint
- LCO - Total number of impact points in the footprint
- IBEGIN - Location in RAIDAT array of first target to be considered for footprinting
- ISTOP - The array location at which all combinations have been tested
- NMAX - Location in RAIDAT array of last target to be considered for footprinting
- IND - A parameter which indicates the availability of more combinations (When exiting, an even value signifies all combinations have been completed)

Common Blocks: MISNR

Subroutines Called: None

Called By: MIRVDUMP

Method:

The candidate targets for a footprint are selected from the n targets starting with location IBEGIN in the RAIDAT array and ending with location NMAX. The number of targets required to complete the footprint (m) is the difference between the total number of impact points in the target (LCO) and the number of distinct impact points in the original footprint (MIRVTGT). Indexes for the targets selected for the trial footprint are stored in the MISNR common blocks. The location of the original distinct impact points are loaded into the first MIRVTGT locations prior to the call to COMBO. COMBO selects the next m targets.

The selection process will be explained by use of an example. Let the input values for MIRVTGT, LCO, IBEGIN, ISTOP, NMAX and IND be 2, 4, 8, 11, 12, and 0 respectively. The even value of IND indicates that this is a new set of targets and initialization is required. The values of K(I), I=3 and 4 (the next two targets starting with the target at the IBEGIN position) are set to 8 and 9 (K(1) and K(2) were set to 1 and 2 prior to the call). IND is set to 1 and a return to the calling program is made. The next call to COMBO is made with no change to the calling parameters or the K array. One is added to the rightmost value of K(I) and the value of 1, 2, 8, 10 are returned. This process continues until the rightmost value of K(I) is equal to NMAX (K = 1, 2, 8, 12). On the next call, the set K = 1, 2, 9, 10 is returned in the K array. The succeeding sets of K returned are 1, 2, 9, 11; 1, 2, 9, 12; 1, 2, 10, 11; and 1, 2, 10, 12. The next combination would be 1, 2, 11, 12, but the value of ISTOP indicates that all combinations of 1, 2, 11, X (where X is any number larger than 11) are not desired. IND is incremented to an even value (2 in this example) which indicates all combinations in this set have been processed.

The flow diagram for COMBO is shown in figure 25.4.

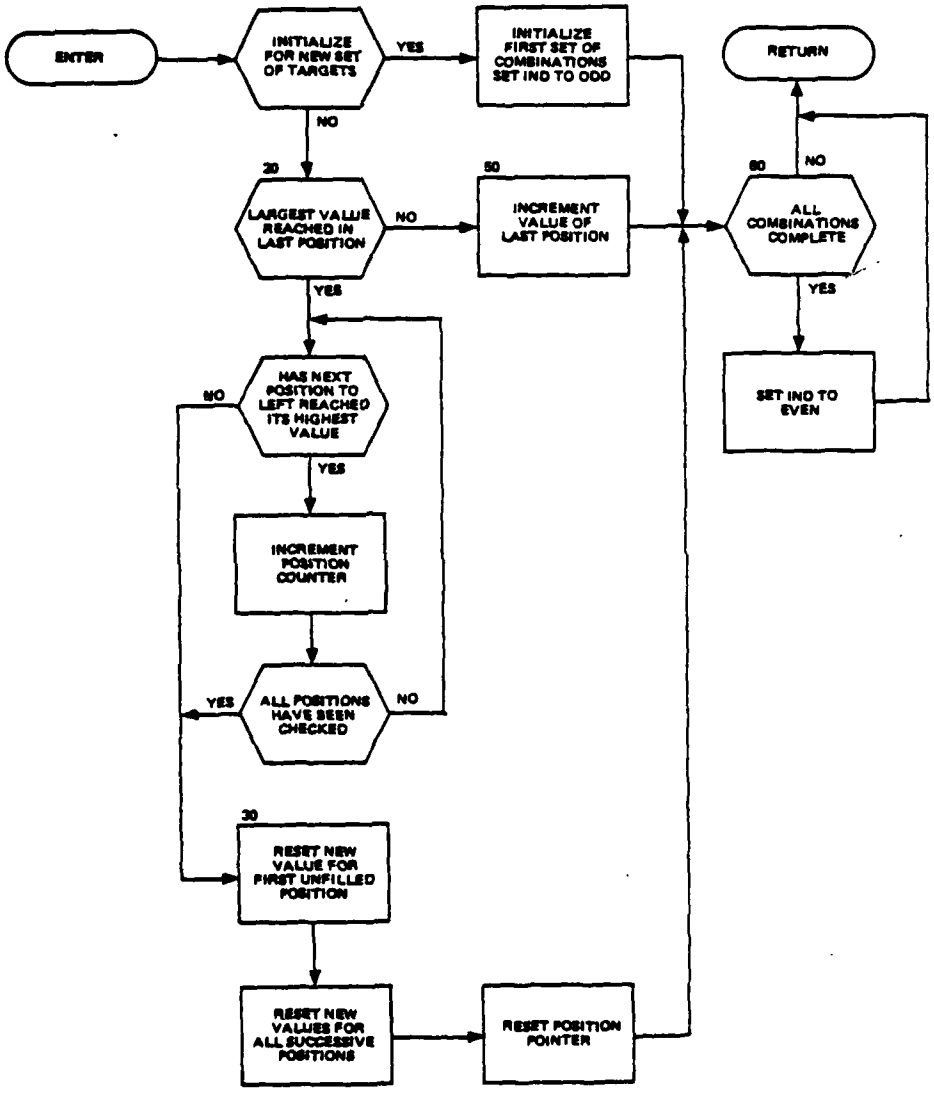


Figure 25.4 Subroutine COMBO

I1.11 Subroutine DRIVER

PURPOSE: DRIVER orders the targets selected for footprint testing by range. It then computes the energy requirements to move the impact from any of the selected targets to any other selected target. Subroutine FOOTCHCK is then called to determine if the selected targets form a feasible footprint.

Entry Points:	DRIVER
Formal Parameters:	IDUMP - Total targets in the footprint
Common Blocks:	EDD, MIST, RAIDAT, RATIO, DSQUARE, MISNR, SYSMAX, EARTH
Subroutines Called:	ORDER, REORDER, FOOTCHCK
Called By:	MIRVDUMP

Method:

The potential targets selected by COMBO are ordered by increasing range to reduce the number of tests required to find a sequence which can be footprinted. The downrange or uprange distance and the crossrange distance from each target to every other target are then calculated. These distances are converted to equivalent downrange distances using uprange and crossrange conversion factors. The resultant equivalent downrange distances are stored in the EDD array. A call to the FOOTCHCK subroutine is then made to determine if the selected set of targets can be footprinted.

The flow diagram for the DRIVER subroutine is shown in figure 25.5.

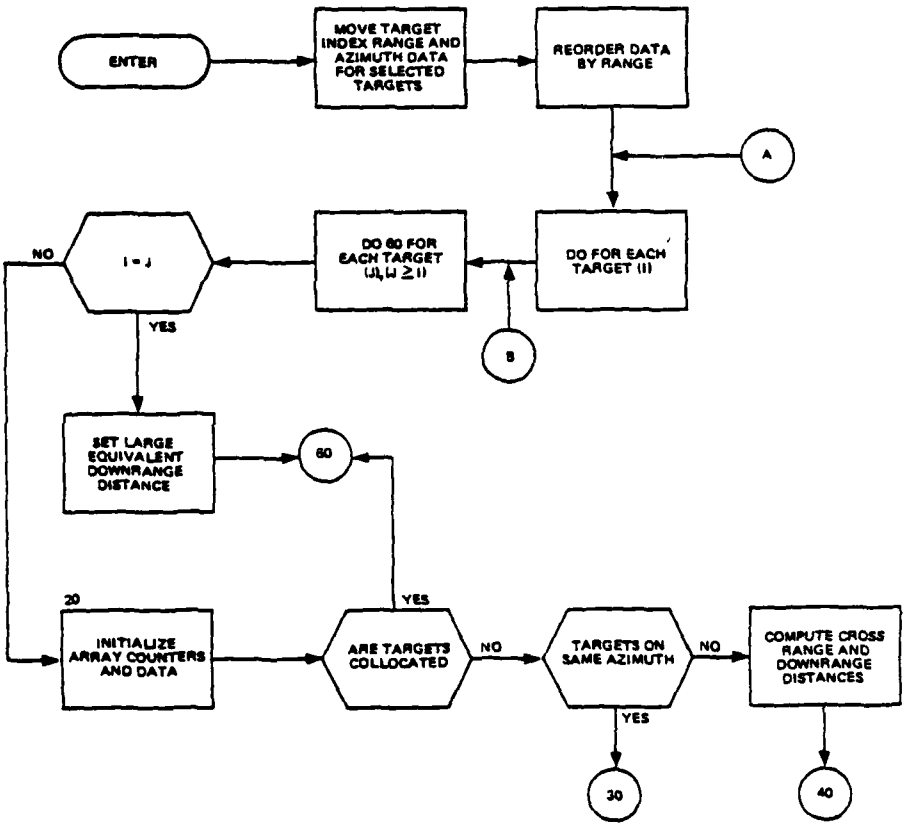


Figure 25.5. Subroutine DRIVER (Part 1 of 2)

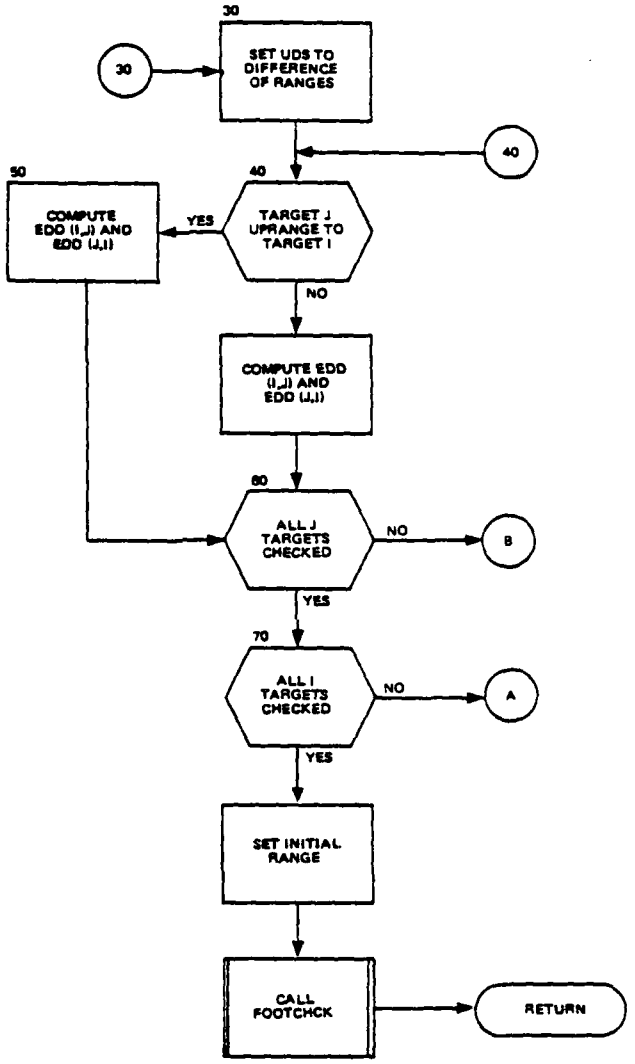


Figure 25.5. (Part 2 of 2)

I1.12 Subroutine FOOTCHCK

PURPOSE: FOOTCHCK processes the targets selected for a potential footprint by COMBO using the effective downrange distances computed in DRIVER and MIRVed missile performance data to determine if the targets can be footprinted. A non-zero value for the variable NHIT indicates that a footprint can be formed.

Entry Points:	FOOTCHCK
Formal Parameters:	None
Common Blocks:	MIST, HIT, SYSMAX, CSAVE, EDD
Subroutines Called:	None
Called By:	DRIVER

Method:

Subroutine FOOTCHCK selects targets from the HOLDING array and determines the fuel required to go from the current target to the next target. The fuel requirement is determined by the effective downrange distance (EDD) between the two targets and the number of reentry vehicles remaining on the MIRV platform. Additional targets are added until the required number have been considered or until all the available fuel has been used. If there is fuel remaining after the required number of targets have been processed, NHIT is set to the number of weapons on the MIRV platform to indicate a footprint is possible. If the fuel is depleted prior to finding a feasible footprint, the targets are checked using a different targeting sequence. When all possible sequences have been tested, a value of zero in NHIT is returned to indicate no footprint is possible.

The flow diagram for the FOOTCHCK subroutine is shown in figure 25.6.

Note: The basic method used in this subroutine is the same as that used in the PATHFIND subroutine of the FOOTPRNT module. The only significant difference is that in FOOTPRNT the first target input must always be the first weapon down. This constraint is imposed to ensure footprinting of fixed targets. In the MIRVDUMP module this constraint is not required.

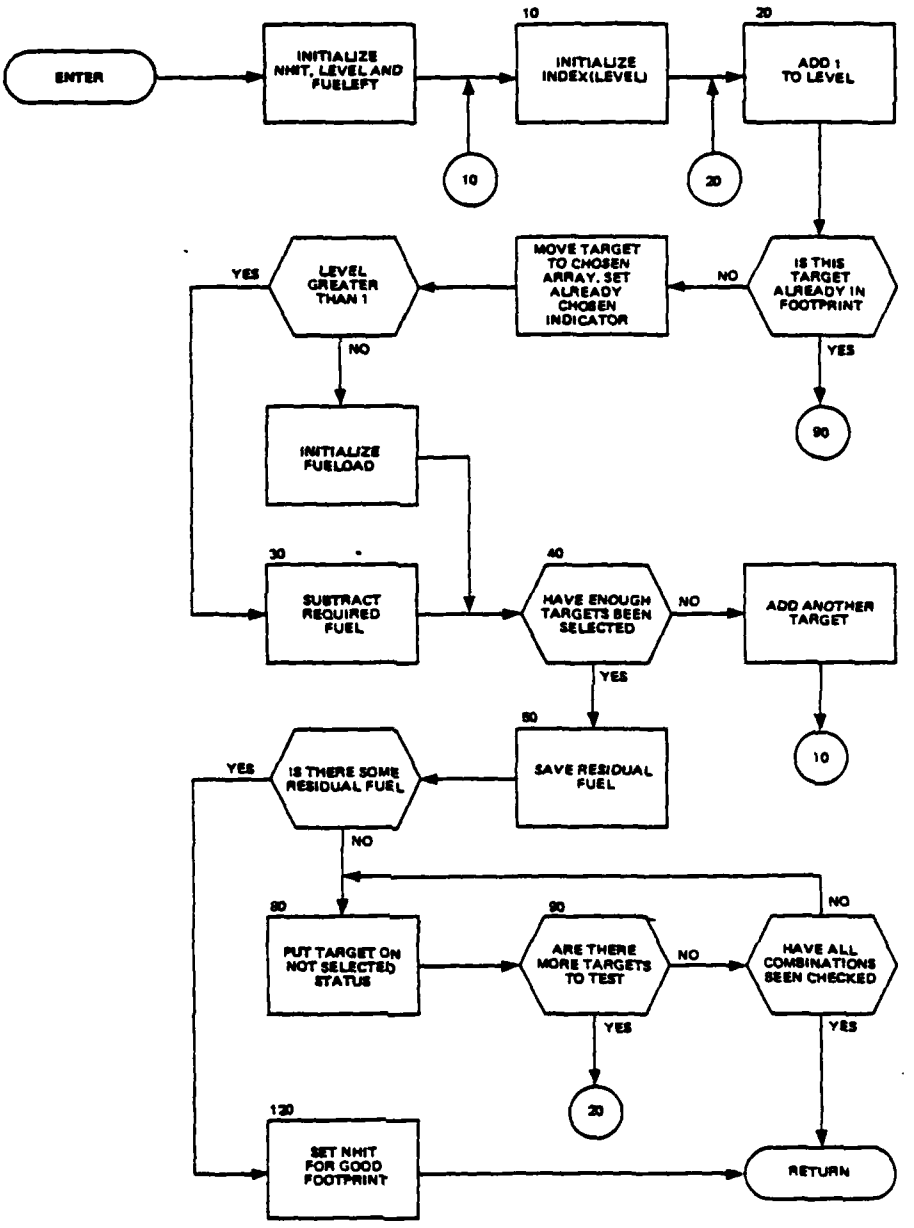


Figure 25.6. Subroutine FOOTCHCK

I1.13 FOOTPRNT Subroutines

The MIRVDUMP module uses three subroutines in the FOOTPRNT module to calculate MIRVed missile performance data. The subroutines are AXES, SETDATA, and XAOS and they are discussed in detail in FOOTPRNT MODULE section of this manual.

THIS PAGE INTENTIONALLY LEFT BLANK

Table 3. (Part 6 of 20)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
DEBUG	IOTA	Index to cell containing Hollerith name of subroutine currently in control
	ICAMFROM(20)	Array containing Hollerith subroutine names in order of calling hierarchy
DRECOV	DRECOV	Distance to recovery
	DSTBC	Distance from launch base to group centroid
EVAL	MINB	The lowest payoff for a bomb in the sortie, found by EVALB
	JDELB	The SORTYTGT index of the bomb with lowest payoff, MINB
	MAXDA	The maximum payoff increment by using ASM on omitted targets
	JADD	The SORTYTGT index of the target with maximum increment MAXDA
	MINDA	The minimum payoff for an ASM in the sortie
	JDEL	The SORTYTGT index of the target for ASM with minimum payoff, MINDA
	MAXOB	The maximum payoff increment for a bomb on an omitted target
	JADDB	The SORTYTGT index of the target showing maximum payoff increment, MAXDAB
	VALSORTY	Extimated total sortie value (=VALDONE (LASTPAY))
	JAF	Index of target to precede insertion
	KALC	Signal to EVAL routines to skip repetition of calculations

Table 3. (Part 7 of 20)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
EVAL (cont.)	VALDIST	"Value" per unit distance of extra range as calculated by FLTPLAN
	VALMAX	Maximum possible value of sortie as currently defined
	JSEQERR	Index of target in sortie showing largest sequence error
	ISFINPLN	Not used
FIXALL	IJFIX(1030) IJFIXR(1030) IKFIX(25)	Logical data for bombers indicating fixed weapon assignment
FIXRANGE	CENTLAT, CENTLONG	Latitude and longitude of centroid of launch bases in group
	DISTC	Distance between centroid of launch bases in group and corridor entry point
FLAG	LXIFLAG(6)	Set to 1 if Ith print is active; 0 if not
FLTPASS	IFPASS	Index of sortie processing pass (limited to 100 sorties per pass)
	JVEHLO	Low-vehicle number, this pass
	JVEHHI	High-vehicle number, this pass
	NVEHPASS	Number of sorties this pass (equals 100, or all remaining sorties)
NPASS	NPASS	Total number of passes necessary
GRPDATA		Characteristics of weapon groups
	IGROUP	Group number
	NWPNS	Number of weapons
	NVEHGRP	Number of vehicles
	IREG	Region
	ITYPE	Weapon type

Table 3. (Part 10 of 20)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
INDEX (cont.)	IDITCH	SORTYTGT index for ditch point (=3)
	IT	Target index
	ICORR	Corridor index
	JTGTIN	SORTYTGT index of the latest target brought in by INPOTGT
IRESRCH	IRESRCH	Flag for subroutine TGTASGN
IS	I1, I2, I3	Temporary storage for input value
KEYS	KEYSTART	Keyword for retrieving number of vehicles
	KEYVBASE	Keyword for retrieving ISTART
NEXTFLT	NTAILS	Number of vehicles assigned to next flight
	JB	Index of launch base of next flight
	SPLIT	Equals 1 if less than 4 bases in group; otherwise =0 (if 1, causes each base to "split"; i.e., send flights down both sides of corridor)
	KB	Marker for side 2
	LB	Marker for side 1
NWPG	NWPG	Total number of weapons in group
NMCLAS		Contains the class name for the NUMTBL record
OUTSRT	IOUTSRT	Sortie index
	MYGROUP	Group index
	MYCORR	Corridor index
	INDVEH	Vehicle index
	JREF	Refuel index
	JDPEN	Depenetration index
	KPAYLOAD	Payload index
	LNCHBASE	Base index
	ITYP	Weapon type
	BASELAT	Base latitude

Table 3. (Part 11 of 20)

BLOCK	VARIABLE OR ARRAY	DESCRIPTION
OUTSRT (cont.)	BASELONG	Base longitude
	NHAP	Number of targets
	HAPTYPE(14)	Type of target
	OBLAT(14)	Latitude of target
	OBLONG(14)	Longitude of target
	DLAT(14)	Latitude of weapon offset
	DLONG(14)	Longitude of weapon offset
	IOBJECT(14)	Index of target
	DSIG(14)	Designator number of target
	TSK(14)	Task and country owner codes of target
	CNTRLC(14)	Country code of target
	FLG(14)	Flag of target
	ATTROUT(14)	Local attrition
	SURVOUT(14)	Cumulative survival probability
	MYHOB	Logical array giving HOB
	DSTLOW1	Low-altitude range (precorridor legs)
	DSTLOW2	Low-altitude range (before first target)
	DSTLOW3	Low-altitude range (after first target)
	SPDLOW	Speed at low altitude
	SPDHIGH	Speed at high altitude
	RANGEX	Range of vehicle without refueling
	DELAY	Delay before takeoff
	IRG	Regional index
	ILRT	Alert status
	IDBOMBER	Bomber identification
	AVLOW	Available low-altitude range

Table 3. (Part 20 of 20)

BLOCK	VALUE OR ARRAY	DESCRIPTION
SORTYTGT	DREC(25)	Distance from depenetration to recovery for target J as last target
	TGTSASGN	Cumulative targets now assigned to raid
TGTASGN	TGTLIM	Value of TGTSASGN not to be exceeded for sortie
	NTGT	Number of targets allocated to raid -- equals NT
	IFSTGT	First target in list to be processed on this call
	ILASTGT	Last target in list to be processed on this call
	IFSTVEH	Index of first sortie to be processed on this call
	ISTVEH	Index of last sortie to be processed on this call
	ISIDE	The corridor side to be flown
VAL	VALRECVR	Ratio of recovery value to total sortie value
	MUSTREC	Parameter care input; if >0 all aircraft must recover
	VUNLOAD	Significance parameter for final alterations in sortie
	VALREC(250)	Ratio of recovery ratio for each weapon group
	TARFAC	User supplied input target multiplier factor
WAROUT	IWARFL	Logical unit number for war gaming print output

3.7 Subroutine ENTMOD

PURPOSE:	To read user's requests and call GETGROUP to control processing
ENTRY POINTS	ENTMOD (first subroutine executed for overlay POST)
FORMAL PARAMETERS:	None
COMMON BLOCKS:	ARRYSIZE, CORRCHAR, C30, DEBUG, IERR, NMCLAS, PAYLOAD, PCALL, POSTHL, WAROUT
SUBROUTINES CALLED:	CINSGET, GETGROUP, INSGET, MODFY, PRNTF, SETFLAG, TIMEME
CALLED BY:	COP

Method:

POSTALOC reads users inputs; stores findings; executes subroutine GETGROUP which controls remaining processing; and prints summary counts.

Subroutine ENTMOD is illustrated in figure 28.

group in order of increasing distance from the corridor entrance. (If weapons from the same group have been previously assigned, bases with remaining aircraft may be mixed with bases from which all aircraft have been assigned; to avoid this, routines which process bases always check numbers of remaining aircraft on each base.) The result, however, is that bases close to each corridor are always processed first (so long as aircraft remain on the bases). In the case of the tactical bombers (JCORR=1 or 2 indicates tactical bombers which use no penetration corridor) or bombers assigned to naval targets, subroutine NOCORR is called to initialize; then CORRPARM is called to order the bases in the same way that the targets will be ordered.

GENRAID next calls CORRPARM to assign values of PHI and RHO to each target and the targets are sorted on the values of PHI.

FLTROUTE is then called to assign the strikes to aircraft beginning with the nearest bases and proceeding until all strikes in the corridor have been assigned. Finally OPTRAID is called to optimize the sorties in the raid.

FLTROUTE and OPTRAID are designed to handle up to minimize computer memory requirements. To account for the unusual occurence of larger raids, a spill provision is included. If FLTROUTE is called for a larger raid, it will return when 100 sorties have been set up. Thus, after OPTRAID, a test is included to be sure all sorties for the corridor have been processed. If not, FLTROUTE is called to continue processing sorties for the corridor and OPTRAID is again called to optimize the sorties. The entrance point for such a continuation is called FLTPASS.

The following points should be noted with regard to subroutine GENRAID:

a. When a penetration corridor is to be used, the launch bases are ordered so that sorties are processed beginning with the nearest base and working back to the farthest. In the case of tactical bombers, the bases are assigned values of RHO and PHI relative to a line between the centroid of the bases and the centroid of the targets. The bases are then arranged in order of PHI.

b. The targets are ordered so that those closest to the corridor origin are hit first. (See Raid Generation in POSTALOC of this manual for further discussion of this.)

c. The current array dimensions limit POSTALOC to processing no more than 100 sorties at once. If more than 100 sorties from a weapon group are to go through the same corridor, it must be done in more than one pass. FLTPA is a second entry point in FLTROUTE, and is used for the second and subsequent passes. NTAILS, a variable usually used to tell FLTROUTE the number of vehicles in the next flight, is set to 1,000 by FLTROUTE to

indicate to GENRAID that another pass is necessary. If NTAILS is negative at the end of FLTROUTE, indicating an error in count of vehicles, it is set to -1000+NTAILS if another pass is called for.

Subroutine GENRAID is illustrated in figure 43.

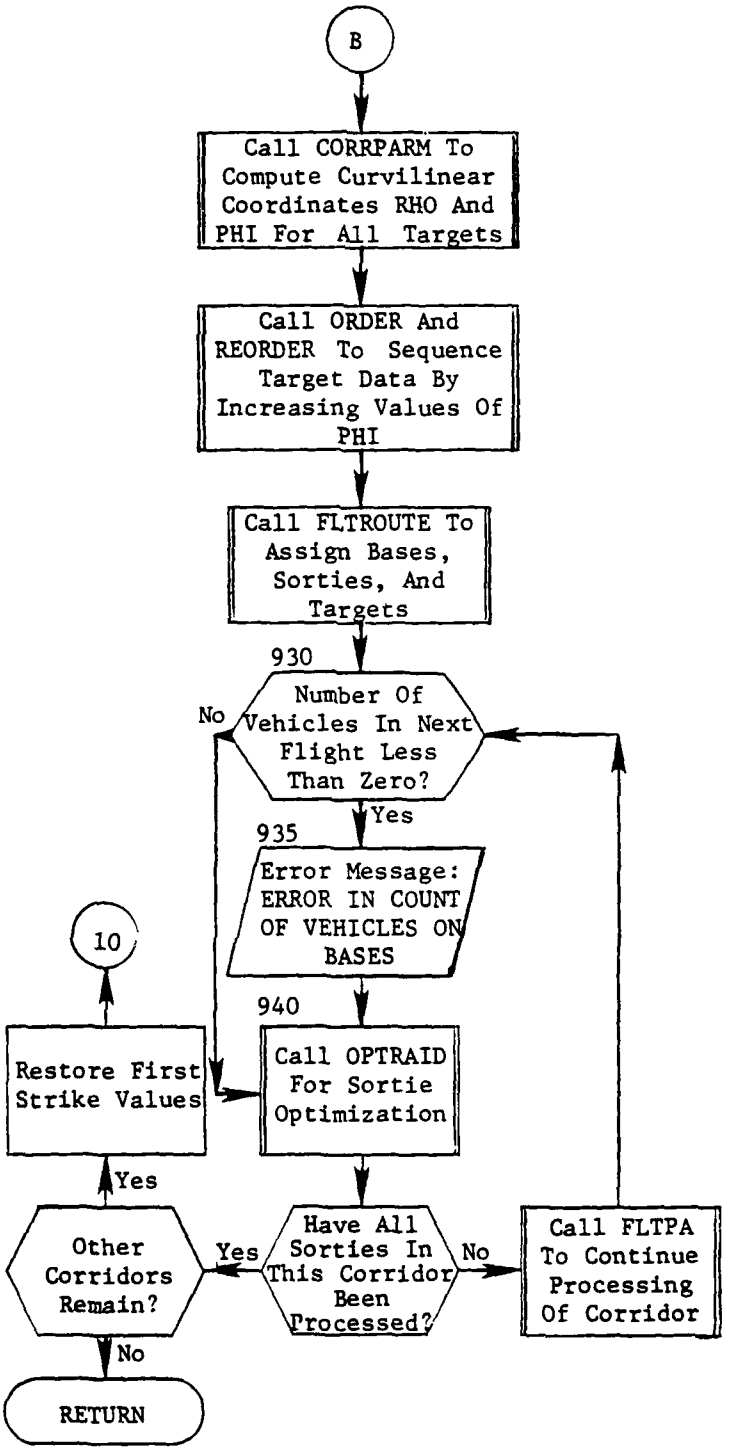


Figure 43. (Part 3 of 3)

3.19 Subroutine GETGROUP

PURPOSE: To read the data base and store parameters necessary for processing and to read each bomber group one at a time and call PRERAID for each group.

ENTRY POINTS: GETGROUP

FORMAL PARAMETERS: None

COMMON BLOCKS: ARAYSIZE, CORRCHAR, C10, C15, C25, C30, CONTROL, DEBUG, DPENREF, GRPDATA, GRPTYP, IDUMP, IFTNO, NMCLAS, PAYLOAD, PCALL, PLANTYPE, POSTHL, PRINTOPT, REFUEL, VAL, WAROUT

SUBROUTINES CALLED: DIRECT, DISTF, DLETE, HDFND, HEAD, ITLE, NEXTTT, OUTSRT, PRERAID, PRINTIT, PRNTF, RETRV, SETFLAG

CALLED BY: ENTMOD (of POSTALOC)

Method:

Subroutine GETGROUP begins by walking the payload chains and storing information for bomber payloads. Next, corridor information is extracted from the data base and stored. This data will be used within the processing scheme. The only remaining data base data necessary for sortie development consists of weapon oriented parameters. GETGROUP walks the WEPGRP chain to query each group one at a time. Only bomber groups are processed where group attribute ATTINC = ATTPOS or all bomber groups when ATTPOS = 0. For each bomber group, weapon group and weapon type attributes are retrieved and stored. Also weapon launch base data associated with the weapon group is retrieved by walking the MYSQDN chain. At this stage all data base entries have been collected and control is passed to subroutine PRERAID for sortie definition.

Subroutine GETGROUP is illustrated in figure 44.

DISTRIBUTION

Addressee	Copies
CCTC Codes	
C124 (Reference and Record Set)	3
C124 (Stock)	6
C126	2
C313	1
C314	7
C630	1
DCA Code	
205	1
EXTERNAL	
Chief, Studies, Analysis and Gaming Agency, OJCS ATTN: SFD, Room 1D935, Pentagon, Washington, DC 20301	2
Chief of Naval Operation, ATTN: OP-654C, Room BE781 Pentagon, Washington, DC 20350	2
Commander-in-Chief, North American Air Defense Command ATTN: NPXYA, Ent Air Force Base, CO 80912	2
U.S. Air Force Weapons Laboratory (AFSC) ATTN: AFWL/SUL (Technical Library), Kirtland Air Force Base, NM 87117	1
Director, Strategic Target Planning, ATTN: (JPS), Offutt Air Force Base, NE 68113	2
Defense Technical Information Center, Cameron Station, Alexandria, VA 22314	12
	42

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENT

This document was prepared under the direction of the Chief for Military Studies and Analyses, CCTC, in response to a requirement of the Studies, Analysis, and Gaming Agency, Organization of the Joint Chiefs of Staff. Technical support was provided by System Sciences, Incorporated under Contract Number DCA 100-75-C-0019. Change set one was prepared under Contract Number DCA 100-78-C-0035. Computer Sciences Corporation prepared change set two under Contract Number DCA 100-78-C-0042.

CONTENTS

Part I

Part II

ILLUSTRATIONS (PART II)

Table 6. (Part 2 of 2)

Tape Name	Field Name	/DEFVAR/Index
ABTAPE(B)	BECM	42
ABTAPE(B)	BWAR	43
ABTAPE(B)	BCRA	44
ABTAPE(B)	BPTC	45
ABTAPE(B)	BTCC	46
ABTAPE(B)	BRFC	47
ABTAPE(B)	BTSK	48
ABTAPE(B)	BHOB	49
ABTAPE(B)	BYLD	50
ABTAPE(B)	BCEP	51
ABTAPE(B)	BCOW	52
STRIKE	SATT	53
STRIKE	SGNM	54
STRIKE	SWNM	55
STRIKE	SLNM	56
STRIKE	SLLT	57
STRIKE	SLLN	58
STRIKE	SLTM	59
STRIKE	SAZM	60
ABTAPE(A)	ABNO	61
ABTAPE(B)	BATT	62

Table 7. PLANOUT Module Internal Common Blocks
(Part 1 of 21)

Block	Array or Variable	Description
ADVRB		The contents of this block are an input clause summary. The variables are either switches which indicate the presence of certain clauses, the "index" or ordinal of the adverb in the "verb adverb array" in the input tables (see MM 9-77, Volume I), a count of a certain type of adverb, or the "pointer" to the beginning of the clause in the input.
	RECSW	True if RECALL mode
	STRSW	True if STRIKE tape is desired
	ABSW	True if ABTAPE is desired
	CHNGSW	True if Sortie Change function is to be used
	ISTRA	Index of first STRIKE SETTING clause
	LSTRA	Index of last STRIKE IF clause
	IABA	Index of first ABTAPE SETTING clause
	LABA	Index of last ABTAPE IF clause
	IFCHNG	Index of first sortie change clause
	LCHNG	Index of last sortie change clause
	IMIST	Index of first MISTME clause
	NMIST	Number of MISTME clauses
	IMCOR	Index of first MSLCOR clause
	NMCOR	Number of MSLCOR clause
	IGTIME	Pointer to GAMETIME clause
	IFUNCO	Pointer to FUNCOM clause
	IONPR	Pointer to ONPRINTS clause

Table 7. (Part 2 of 21)

Block	Array or Variable	Description
ARTIME	ARTIME(50)	Earliest bomber arrival time at refuel area I
	NBUDREF	Number of "buddy" refuelings required
	NBOMBREF(50)	Number of bombers assigned to refuel area I
	NTANKREF(50)	Number of tankers assigned to refuel area I
	IARVLS/ARVLS(2,1000)	ARVLS(1,I) = time of the Ith bomber refuel processed by PLNTPLAN; IARVLS(2,I) = the refuel area for that bomber refuel
ASMARRAY	ALAT(14)	Aim point latitude
	ALON(14)	Aim point longitude
	IFLY(14)	Fly point flag
	IDIS(14)	Distance from fly point to ASM target
	IORD(14)	Sort index
	JAY	Index communicated to PREFL1, PREFL2
	DIST	Distance communicated to PREFL1, PREFL2, POSTFLY
CALLSW		Each switch in this block is set to true after the indicated subroutine is executed. Its purpose is to prevent subsequent executions
	WDSW	Switch for WEPDATA
	CORSW	Switch for GEOGET
	LNDSW	Switch for LNCHDATA
CONTR1*	ICHANGE	Switch for FLTSORT

*The following variables in this common block are no longer used: LREAD(2), MBOM, MMIS, IST, LTST, INDSK, IOUTDSK, INDSK, INDSKM, and IOUTDSKM.

Table 7. (Part 3 of 21)

Block	Array or Variable	Description
CONTROL	LSIDE	Side whose sorties are being used (1 = BLUE, 2 = RED)
CORCOUNT	IH	Points to line of /HAPPEN/ where current corridor begins
	KC	Number of lines in /HAPPEN/ describing current corridor
	JH	Points to line of /HAPPEN/ where current depenetration corridor begins
	LC	Number of lines in /HAPPEN/ describing depenetration corridor
CORRC1	MCORCH1	Maximum number of penetration corridors
	IDEFDST(30)	Total precorridor defended distance
	IMPRTD(30,3)	Order of importance of attrition per nautical mile
	ATPDST(30,3)	Average attrition per nautical mile in the Jth corridor, Ith leg
CORRCHAR		Penetration corridor characteristics
	PCLAT(30)	Orientation point latitude
	PCLONG(30)	Orientation point longitude
	RPLAT(30)	Origin latitude
	RPLONG(30)	Origin longitude
	ENTLAT(30)	Entry latitude
	ENTLONG(30)	Entry longitude
	CRLENGTH (30)	Distance from entry to origin
	KORSTYLE(30)	Parameter to adjust mode of corridor penetration

Table 7. (Part 4 of 21)

Block	Array or Variable	Description
CORRCHAR	ATTRCORR(30)	High-altitude attrition per nautical mile, unsupressed
	ATTRSUPF(30)	High-altitude attrition per nautical mile, suppressed
	HILOATTR(30)	Ratio low- to high-altitude attrition (less than one)
	DEFRANGE(30)	Characteristic range of corridor defense (nautical miles)
	NPRCRDEX(30)	Number of attrition sections
	DEFDISTX(30,3)	Length of attrition section
	ATTRPREX(30,3)	Attrition per nautical mile of attrition section
DATEOFC	CDAT(14)	Values to be printed as CHANGE DATE in Bomber and Missile detailed plans
	NDATA	Number of corridors
DECA	DELDIS(6)	Decoy coverage distance
	LPRIORITY(20)	Possible decoy launch priority
	LMHT(90)	Possible decoy launch event number
	NDCYRQ(20)	Pointer to array DELDIS
	NPSLN	Number of possible decoy launches
	NUMDCOYS	Number of decoys available
DINDATA	HDT(90)	Time -- for detailed history -- of event I
	KPL(90)	Place of event I
	JTP(90)	Event type of event I
	HLA(90)	Latitude of event I
	HLO(90)	Longitude of event I
	TZT(90)	Weapon offset latitude of event I

Table 7. (Part 5 of 21)

Block	Array or Variable	Description
DINDATA (cont.)	TZN(90)	Weapon offset longitude of event I
	PA(90)	Probability of arrival at target of event I
	MHT	Total number of lines in detailed plan
	NPL	Number of planned events
DINDT2	CMT(90)	Cumulative time of event I
	IWH(90)	Warhead type index of event I
DISTC	DISTC(20)	Distances between target events
DPENREF	DPLAT(50)	Depenetration latitude
	DPLONG(50)	Depenetration longitude
	QFLAT/RFLAT(20)	Refuel point latitude
	QFLONG/RFLONG(20)	Refuel point longitude
EVCOM	TELAPSE	Elapsed time to current event
	ICEV	Event count to current event
	EVLAT	Latitude of event
	EVLONG	Longitude of event
EVENTS	LAUNM	Missile launch code
	LAUNB	Bomber launch code
	LEREFUEL	Refuel code
	LOCLATTR	Local attrition or drop bomb event code
	LAUNASM	Launch ASM event code
	LAUNDCOY	Launch Decoy event code

Table 7. (Part 6 of 21)

Block	Array or Variable	Description
EVENTS (cont.)	LANDHO	Recovery event code
	LOHI	Change Altitude event code
	MISSATTR	Missile attrition event code
	LEGDOG	Dogleg event code
	LABORT	Abort event code
	LENTEREF	Enter refuel area event code
	LEAVEREF	Leave refuel area event code
	IGOHI	Go to high altitude event code
	IGOLOW	Go to low altitude event code
GAMETIME	KDAY	Day of game
	KMON	Month of game
	KYEAR	Year of game
	HHR	H-Hour
GRPSTF	IPAY(250)	Weapon group payload index
	ITYPEX(250)	Weapon group type index
	IREGON(250)	Weapon group region
	GSBLX(250)	Weapon group prelaunch survival probability
	GPKNA(250)	Single shot kill probability against naval targets for weapon group I
	IGCLS(250)	Weapons group class index
	IGLERT(250)	Weapon group alert status

Table 7. (Part 7 of 21)

Block	Array or Variable	Description
HAPPEN	KOUNT(30)	Number of /HAPPEN/ lines for penetration corridor
	IHAP(30)	Pointer to first line entry for penetration corridor
	MOUNT(50)	Number of /HAPPEN/ lines for depenetration corridor
	JHAP(50)	Pointer to first line entry for depenetration corridor
	JAPTYPE(250)	Attrition section indicator (1,2,3 enter section; 4,5,6 leave section)
	HAPLAT(250)	Latitude of corridor point
	HAPLONG(250)	Longitude of corridor point
	HAPDIST(250)	Distance from previous point
HILO	ISTOREHI	Number of events in /OUTSRT/ after which GO HIGH occurs
	ISTORELO	Number of events in /OUTSRT/ after which GO LOW occurs
	IGOLEFT	Set to 1 if GO LOW range is available after depenetration
	FACHI	Distance after event ISTOREHI at which GO HIGH is located
	FACLO	Distance after event ISTORELO at which GO LOW is located
	GOLO	Amount of GOLOW range remaining for depenetration
ICLASS	IBOMBER	Bomber class index
	ITANKER	Tanker class index

Table 7. (Part 8 of 21)

Block	Array or Variable	Description
IDP	IDP(2)	Depenetration corridor index number as reassigned when last target is an ASM target
IFSCOM	ISTPAR	Number of SETTING/IF clause pairs for STRIKE tape
	JSIF(100)	Pointers to IF clauses for STRIKE tape
	JSSET(100)	Pointers to SETTING clause for STRIKE tape
	IABPAR	Number of SETTING/IF clause pairs for ABTAPE
	JABIF(100)	Pointers to IF clauses for ABTAPE
	JABSET(100)	Pointers to SETTING clauses for ABTAPE
IFUNC	JFUNC(20)	Function codes input (alphabetic)
	INDFUNC(20)	Numeric function codes
IGO	IGO800	Set to 1 for degenerate target area
INDATA	INDATA	Side (1=Blue, 2=Red)
	INBASE	Launch base index
	INDV	Vehicle index
	ILAST	Number of MIRVs per missile
	ITYPE	Weapon type index
	ICLZSS	Weapon class index
	IRZG	Weapon group region
	IZLERT	Weapon group alert status

Table 7. (Part 9 of 21)

Block	Array or Variable	Description
IOUT	LPAYLOAD	Index of current payload
	LREF	Index of current refuel area
	LDPEN	Index of current depenetration point
	KOKO	Index of current ASM type
	JFCTNO	Function code of current vehicle
IRF	IRF	Assigned refuel area index
	NRF	Number of refuel areas
LASM	U1	Latitude of beginning point of bomber path
	V1	Longitude of beginning point of bomber path
	U2	Latitude of end point of bomber path
	V2	Longitude of end point of bomber path
	UAT	Latitude of ASM target
	VAT	Longitude of ASM target

Table 7. (Part 10 of 21)

Block	Array or Variable	Description
LASM (cont.)	RASM	Range of ASM
	RLAT	Latitude of ASM aim point
	RLONG	Longitude of ASM aim point
LASREF	LASREF	Reference Code (IDS) of last non-tanker sortie
LAUNSNAP	INRANGE	Set to zero if ASM target is in range of flight path; otherwise to one
	FRACPATH	Fraction of total path at which ASM is launched
MH2	MHMIN(2)	Lower plot markers for sortie
	MHMAX(2)	Upper plot markers for sortie
MISCT	MISCT	Missile booster count
	MTARGCT	Missile target count
MODE	MODE	1 for high altitude, 4 for low altitude
NOFSYS	NOFSYS(100)	Offensive system type index

Table 7. (Part 11 of 21)

Block	Array or Variable	Description
OUTSRA	INDR(10)	Change indicator for targets
	DLTA(10)	Change in time information
	ICTIME	Change time flag
OUTSRT	ISORTN	Sortie Number
	IOUTSRT	Sortie Number
	MYGROUP	Weapon group index
	MYCORR	Penetration corridor index
	INDVEH	Vehicle index
	IREF	Refuel index
	IDPEN	Depenetration corridor index
	IPAYLOAD	Payload table index
	LNCHBASE	Launch base index number
	ITYP	Weapon group type index
	BLAT	Launch base latitude
	BLONG	Launch base longitude
	NHAP	Number of sortie events
	IBTYPE(14)	Event type
	OBLAT(14)	Latitude of event
	OBLONG(14)	Longitude of event
	DLAT(14)	Target offset - in degrees latitude
	DLONG(14)	Target offset - in degrees longitude

Table 7. (Part 12 of 21)

Block	Array or Variable	Description
OUTSRT (cont.)	IBJEC(14)	Event place code
	IBDES(14)	Target DESIG
	IBTSK(14)	Task code of target
	IBCTY(14)	Country code of target
	IBFLG(14)	Target FLAG
	ATTROUT(14)	Local attrition
	SURVOUT(14)	Cumulative survival probability
	LXMYHOB(1)	Height of burst indicator
	GOLOW1	Low altitude range available for use in corridor
	GOLOW2	Low altitude range available for use before first target
	GOLOW3	Low altitude range available for use after first target
	SPDLQ	Speed at low altitude
	SPDHI	Speed at high altitude
	RANGX	Range of vehicle
	RANGREF	Refueled range of vehicle
	DELAY	Delay of vehicle launch
	IRG	Vehicle launch region
	ILRT	Vehicle alert status
	IDBOMBER	Vehicle identification
	AVAILOW	Available low altitude range

Table 7. (Part 13 of 21)

Block	Array or Variable	Description
OUTSRT (cont.)	RNGDEC	Range decrement at low altitude
	DRECOVER	Distance to recovery
	DISTLEGO	Distance to origin
PAYSTF	NOBOMB1(40)	Number of bombs of type 1 (number of RVs for missiles)
	IWHD1(40)	Warhead index of bomb type 1
	NOBOMB2(40)	Number of bombs of type 2
	IWHD2(40)	Warhead index of bomb type 2
	NASM(40)	Number of ASMs
	IASM(40)	Warhead index of ASM
	NPCM(40)	Number of counter measures
	NPDCY(40)	Number of decoys
	NAPDCY(40)	Number of area decoys
	IMIRV(40)	MIRV system identifier
	PYALT(40)	Weapon release altitude (bombers)
	PNAME(100)	Payload name
	NPAX	Number of payloads
POLITE	S1	Latitude of beginning interpolation point
	T1	Longitude of beginning interpolation point
	S2	Latitude of interpolation end point
	T2	Longitude of interpolation end point

Table 7. (Part 14 of 21)

Block	Array or Variable	Description
POLITE (cont.)	FACTOR	Interpolation factor or fraction
	SR	Latitude of interpolated point
	TR	Longitude of interpolated point
PPINFO	NDUMCORR	Tactical aircraft corridor index
	GOLOX1	Saved low altitude range available for use in corridor
	GOLOX2	Saved low altitude range available for use before first target
	GOLOX3	Saved low altitude range available for use after first target
	INDEX	Bomber plan index equals group number plus 100 times corridor index plus 10000 times sortie number
	NSORTIES	Total number of bomber plans processed
	INDXX	Group weapon type index
	GOGO	Saved low altitude range available for use in corridor
	MHIST	Maximum number of entries into history table
	DUST	Distance bomber traveled during first history event
	LXIDPCHK(2)	Logical area indicating if depenetration corridor is used.
PPXX	ILAUNDEX(90)	Number of decoys launched
	TIMELAUN(90)	Time of decoy launch
	DISTORE(90,6)	Distance traveled by decoy
	HDTX(90)	Temporary line array

Table 7. (Part 15 of 21)

Block	Array and Variable	Description
PPXX (cont.)	KPLX(90)	Temporary place array
	JTPX(90)	Temporary event number array
	HLAX(90)	Temporary latitude array
	HLOX(90	Temporary longitude array
	TZTX(90)	Temporary offset latitude array
	TZNX(90)	Temporary offset longitude array
	IWHX(90)	Temporary warhead index array
	PAX(90)	Temporary probability of arrival array
	CMTX(90)	Temporary cumulative time array
PREVNT	PRLAT	Latitude of previous event
	PRLONG	Longitude of previous event
PRNCON	INDEXPR(15)	Print request number
	JAGROUP(15)	First group for request
	JACORR(15)	First corridor for request
	JSSORT(15)	First sortie for request
	LAGROUP(15)	Last group for request
	LACORR(15)	Last corridor for request
	LASORT(15)	Last sortie for request
	KFREQ(15)	Frequency for request
	NREQ	Number of requests
PSW	STPRIN	True if STRIKE tape print requested
	ABPRIN	True if ABTAPE print requested
	ABUNIT	Report code for ABTAPE print

Table 7. (Part 16 of 21)

Block	Array or Variable	Description
RECBAS	RCBLAT(50,4)	Recovery base latitude*
	RCBLON(50,4)	Recovery base longitude*
	INDBAS(50,4)	Recovery base name*
	INDCAP(50,4)	Recovery base capacity*
	DISTR(50,4)	Distance to recovery*
	TOF(50,4)	Time of flight to recovery*
RECOVERY	NAMECAP(200,3)	J=1, Recovery base name J=2, Recovery base capacity J=3, Number of aircraft that launched at recovery base
	NUSED(200)	Working array that defines the number of aircraft arriving at each base for a given group
	IREC	Logical unit containing recovery base data
	NREC	Number of unique recovery bases
	NBOMGP	Number of bomber groups
	NMSGRP	Number of missile groups
RL	RL	Decoy low-altitude range
	RH	Decoy high-altitude range
SNAPON	NAP(15)	Set to three for active print I; set to one for inactive print I

* Indexed for each of four bases assigned to each depenetration point.

Table 7. (Part 17 of 21)

Block	Array or Variable	Description
TANKA	IINDEXTK(60)	Tanker base index
	TKRLAT(60)	Latitude of tanker base
	TKRLONG(60)	Longitude of tanker base
	IIREFTK(60)	Refuel area for tankers where N > 0 implies must refuel at area n
	NTKPSQN(60)	Number of tankers in squadron at base
	NALRTNK(60)	Number of alert tankers at base
	TANKSPP(60)	Speed of tankers at base
	TKDLYALT(60)	Delay for alert tankers at base
	TKDLYLN(60)	Delay for non-alert tankers at base
	TKTTOS(60)	Total time on station
	IITYPTK(60)	Tanker type index
	TRANGX(60)	Tanker range
TANKB	COST(60,50)	Distance between tanker base I and refuel area J
	SOURCE(60)	Number of tankers at base I to be automatically assigned
	ISOL(110)	The Ith nonzero element in the final VAM solution
	RBASLOC(110)	Tanker base corresponding to the Ith solution element
	CBASLOC(110)	Refuel area corresponding to the Ith solution element
	NSOL	Number of nonzero elements in VAM solution

Table 7. (Part 18 of 21)

Block	Array or Variable	Description
TANKB (cont.)	RMAX	Number of rows (tanker bases) in VAM problem
	LXIRCHK(2)	Logical array true if base not automatically allocated
	IRCDIF	Number of bases for which LXIRCHK is true
	DISTREF(50)	Distances from current tanker base to refuel area I
TANKER	INDEXTK	Tanker index
	TKLAT	Tanker latitude
	TKLONG	Tanker longitude
	IREFLK	Tanker refuel area index
	NPSQNTK	Number of tankers per squadron
	NALRTK	Number of alert tankers
	SPEEDTK	Tanker speed
	DLYALTK	Delay for alert tankers
	DLYNLTK	Delay for nonalert tankers
	TTQS	Total time on station
	ITYPETK	Tanker type index
	TANKRNGE	Tanker range
TEMPO	DT(50)	Distance or time temporary storage
	JT(50)	Event type temporary storage
	TLT(50)	Latitude temporary storage
	TLN(50)	Longitude temporary storage
	LPL(50)	Place index temporary storage

Table 7. (Part 19 of 21)

Block	Array or Variable	Description
TIMELINE	LXITIM(2)	Packed logical array: True if type of CORMSL is "line"
	CORMSLX(40)	Percent flight complete or time on line
	ZLAT(50,2)	Latitude of timing end points
	ZLONG(50,2)	Longitude of timing end points
	XC(50)	X-coordinate of cross product vector of lining line
	YC(50)	Y-coordinate of cross product of timing line
	ZC(50)	Z-coordinate of cross product of timing line
	DL(50)	Length of timing line
	NLINES	Number of timing lines
TSTUFF		Communication with TOFM
	XTOFMIN	Minimum time of flight
	XCMISS	Missile flight parameter
	XRNGMIN	Minimum range
	XRANGE	Maximum range
TYPSTF		Weapon type data
	TRANGE(100)	Weapon range
	TCEP(100)	Weapon CEP at zero range
	TTOFMIN(100)	Weapon minimum time of flight
	TCMISS(100)	Weapon missile constant
	TRNGMN(100)	Weapon minimum range

Table 7. (Part 20 of 21)

Block	Array or Variable	Description
TYPSTF (cont.)	TREL(100)	Weapon reliability
	TNAME(100)	Weapon name
	TFUNCT(100)	Weapon function code
	TLINT(100)	Weapon launch interval
	ISMLUN(100)	Number of weapons which may be launched simultaneously
	TSLOP(100)	Weapon CEP equation slope
VICINITY	VHB	Bomber cannot go high within VHB miles before target
	VHA	Bomber cannot go high within VHA miles after target
	VLB	Bomber cannot go low within VLB miles before target
	VLA	Bomber cannot go low within VLA miles after target
	GOMIN	Bomber cannot fly low for less than GOMIN minutes
WEPTRN	IWREG	Launch Region
	WCOW	Launch Country Location
	IERT	Record Type
	WBENO	Launch Base BE Number
	WLNNM	Launch Base Name
	TBLN	Elapsed Time To Launch - Bombers Only

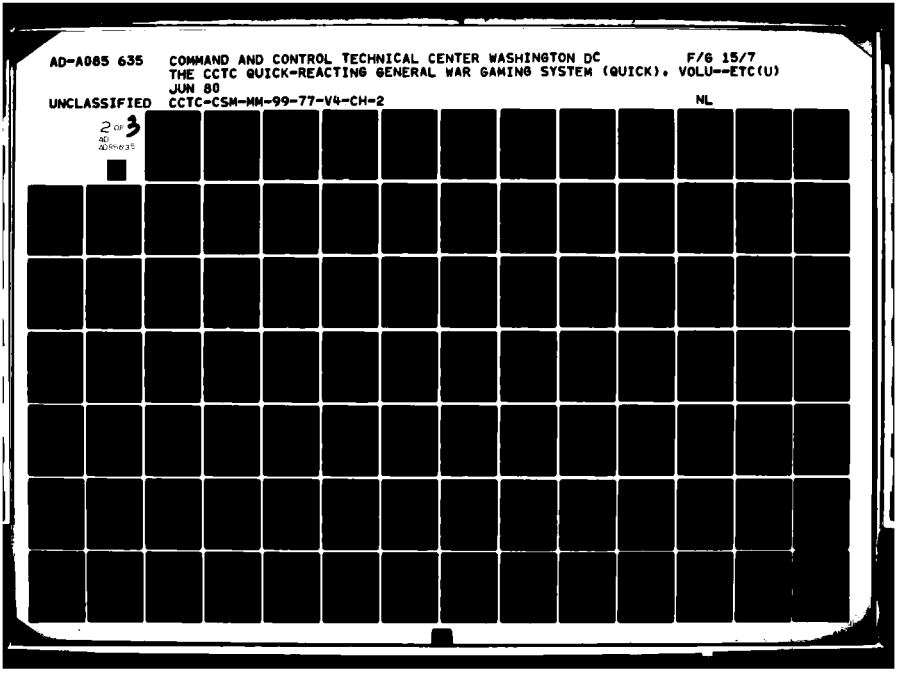
Table 7. (Part 21 of 21)

Block	Array or Variable	Description
WHDSTF	WHDYLD(50)	Warhead yield
	WHDFRC(50)	Warhead fission/fusion fraction
	WHDRNG(50)	Warhead range*
	WHDREL(50)	Warhead reliability*
	WHDCEP(50)	Warhead CEP at zero range*
	WHDSPD(50,	Warhead speed*
	WHDSLP(50)	Warhead CEP equation slope*

* ASM warhead only

THIS PAGE INTENTIONALLY LEFT BLANK

AD-A085 635 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 15/7
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK). VOLU--ETC(U)
JUN 80
UNCLASSIFIED CCTC-CSM-MM-99-77-V4-CH-2 NL



4.7 Subroutine ENTMOD

PURPOSE: Driver subroutine for PLANOUT Module

ENTRY POINTS: ENTMOD (first subroutine called when overlay PLANOUT is executed)

FORMAL PARAMETERS: None

COMMON BLOCKS: ADVRB, C15, C30, CALLSW, LASREF, PRNCON, ZEES

SUBROUTINES CALLED: ABORT, ALTPLAN, HDFND, INITANK, INSGET, INTRFACE, PLANTANK, PLNTPLAN, RETRV

CALLED BY: MODGET

Method:

First the switches in the /CALLSW/ block are set to false. These switches prevent the routines GEOGET, LNCHDATA and WEPDATA from being called more than once. Next the NUMTBL record is retrieved. The reference code of the last nontanker sortie (LASREF) is obtained by cycling the SORTIE chain and saving the reference code of the sortie unless it is a tanker sortie.

Now all adverbs are read and switches and pointers in block /ADVRB/ are set accordingly. The ONPRINTS clause is now processed to set values in the /PRNCON/ block.

Finally, the appropriate overlays are read in and executed according to switches set by adverbs. If RECALC was input, INITANK, PLNTPLAN and PLANTANK are called. IF ACARD, CCARD or ICARD clauses were included, ALTPLAN is called. And, if STRIKE or ABTAPE were input, INTRFACE is called.

Subroutine ENTMOD is illustrated in figure 62.

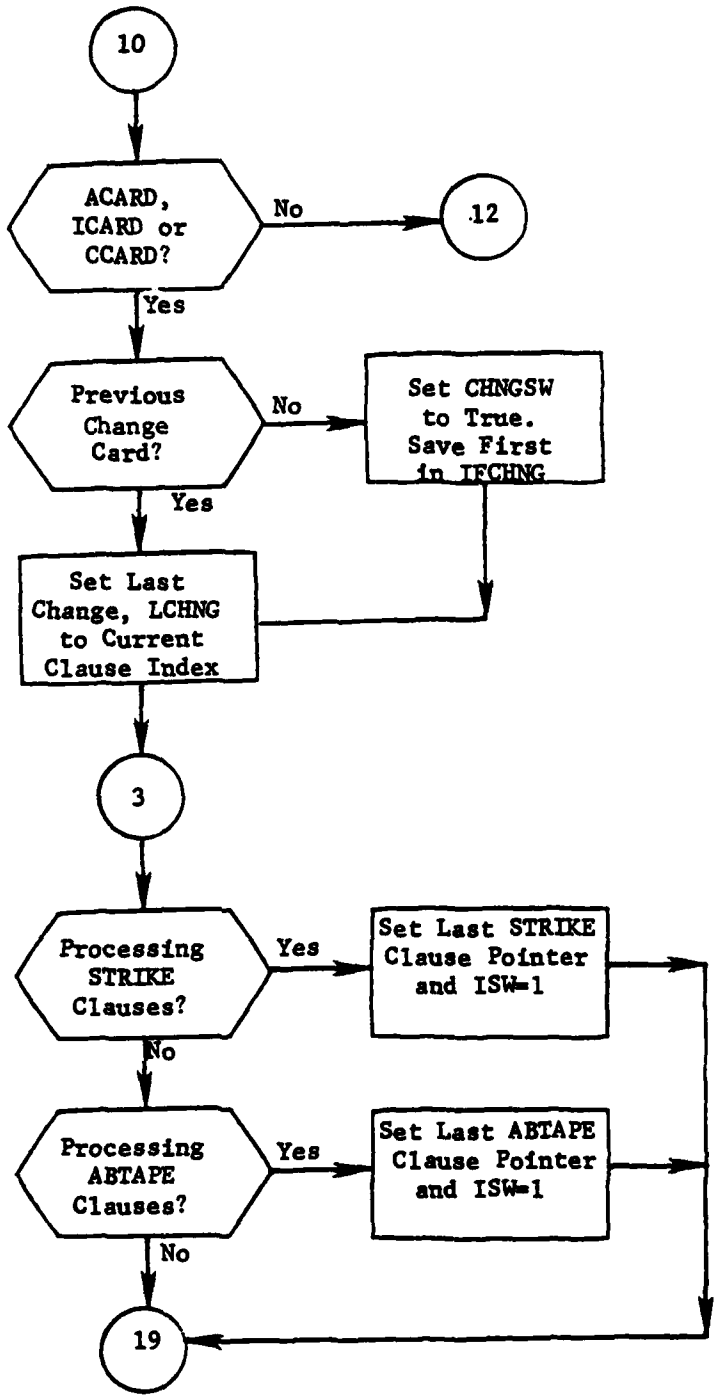


Figure 62. (Part 5 of 13)

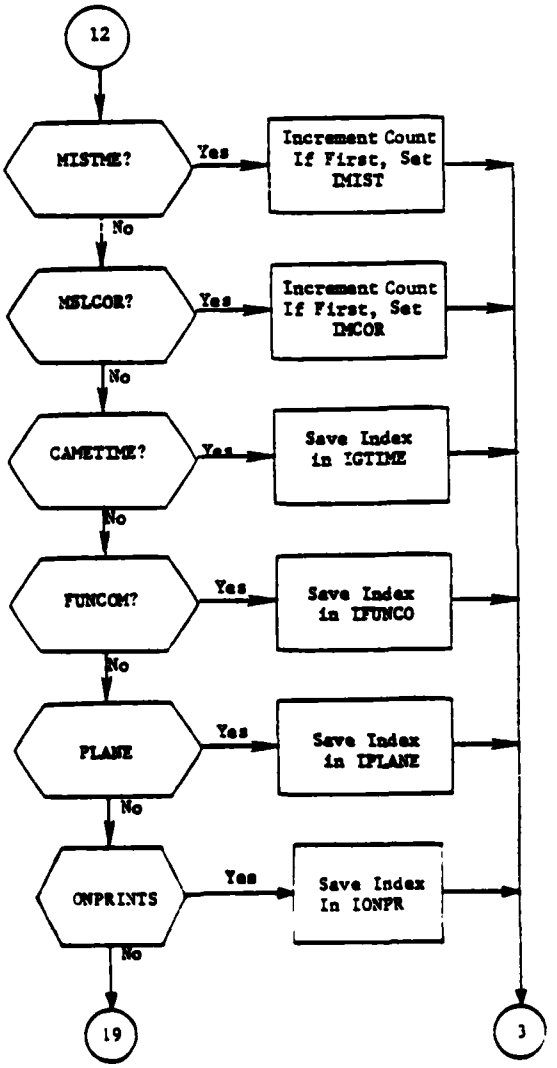


Figure 62. (Part 6 of 13)

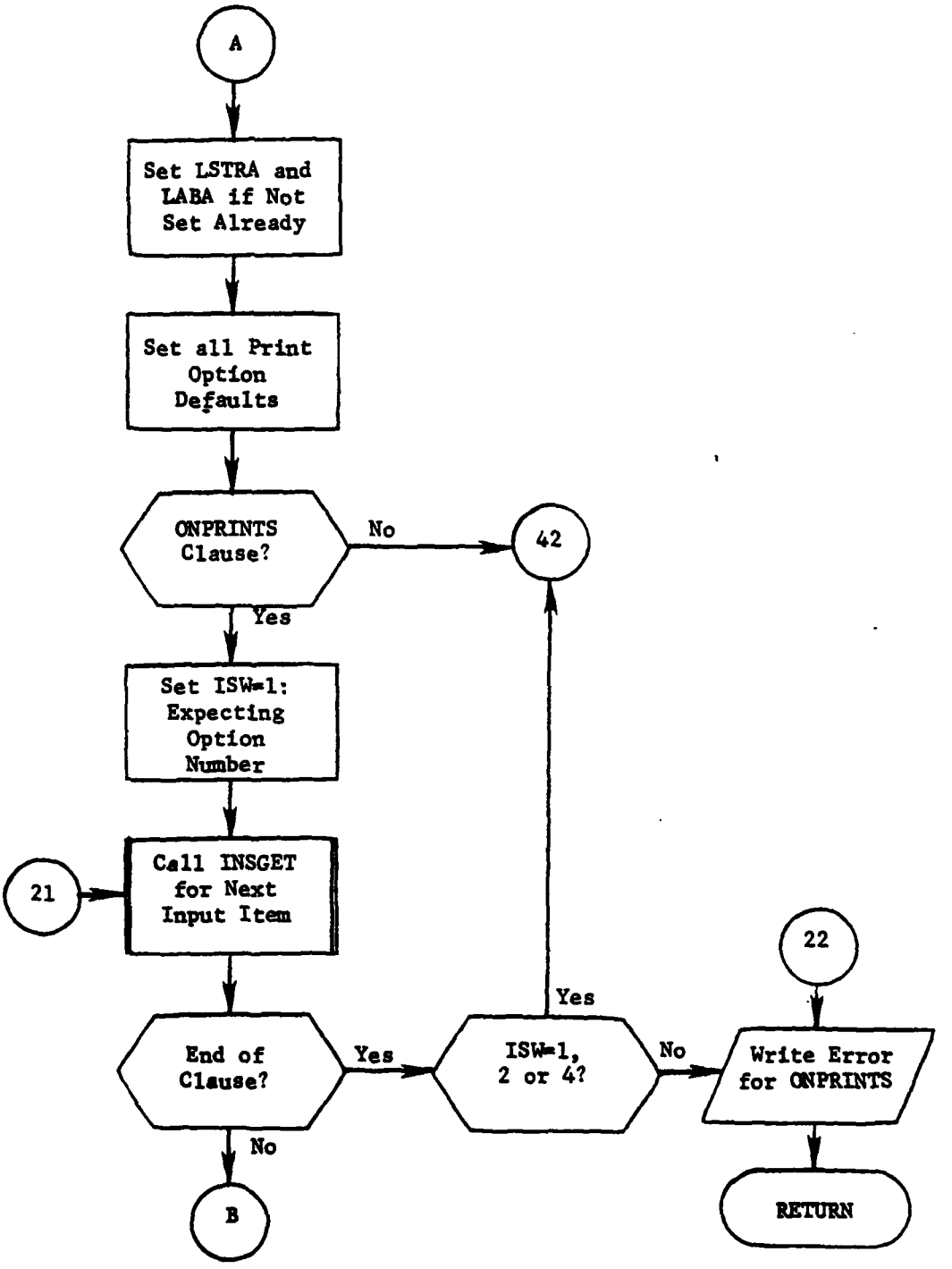


Figure 62. (Part 7 of 13)

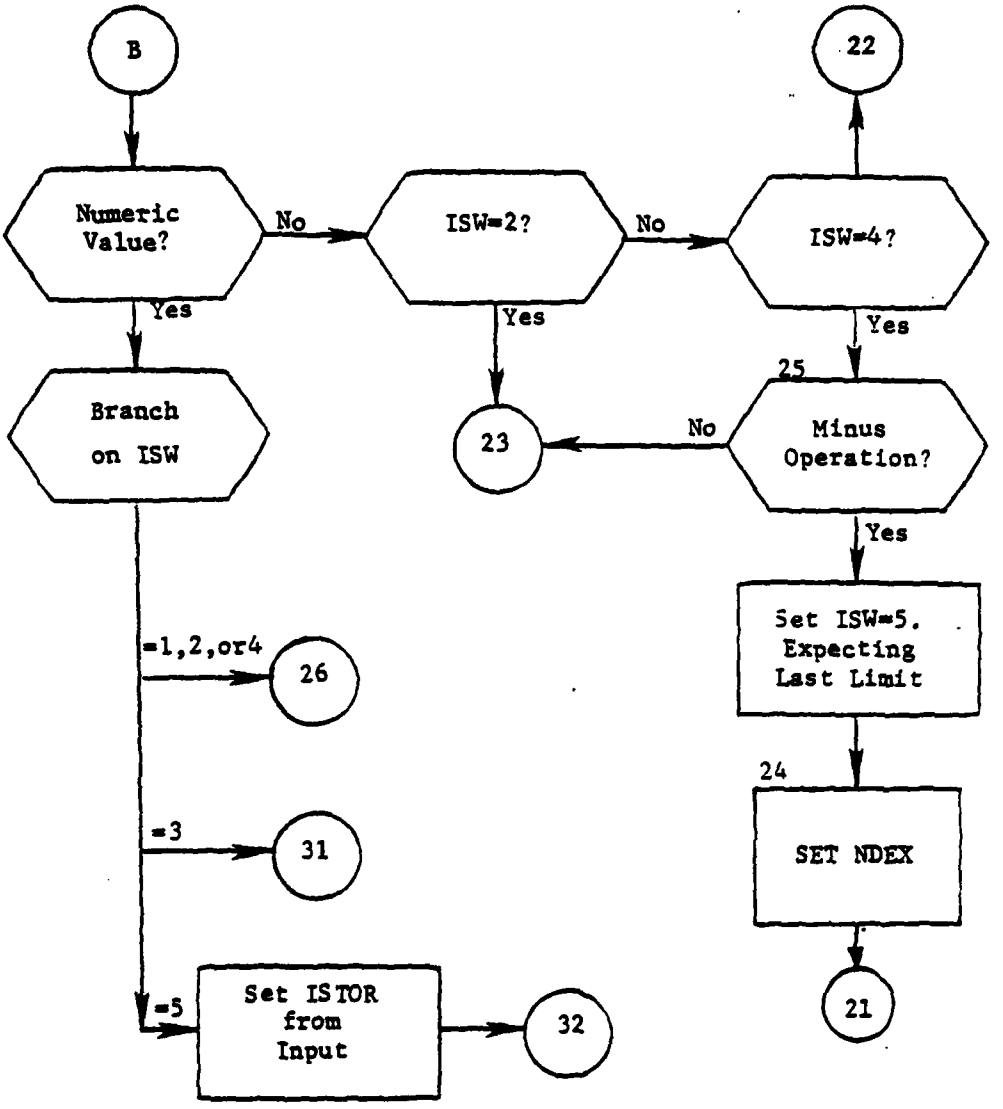


Figure 62. (Part 8 of 13)

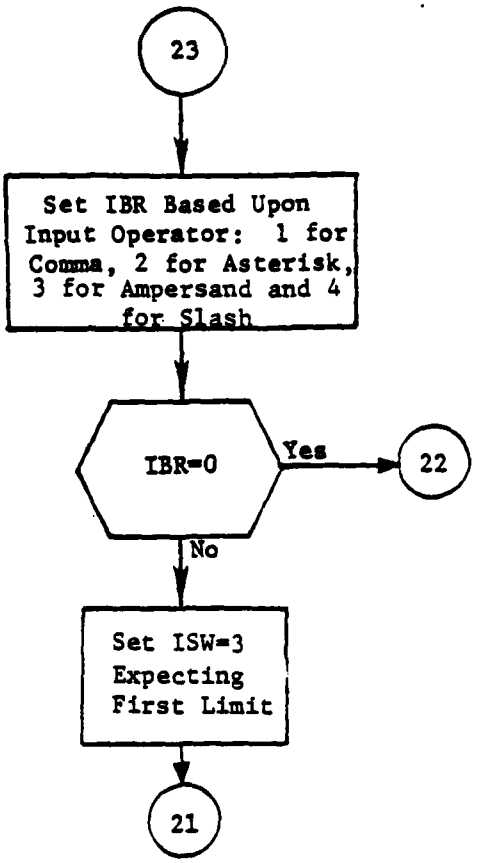


Figure 62. (Part 9 of 13)

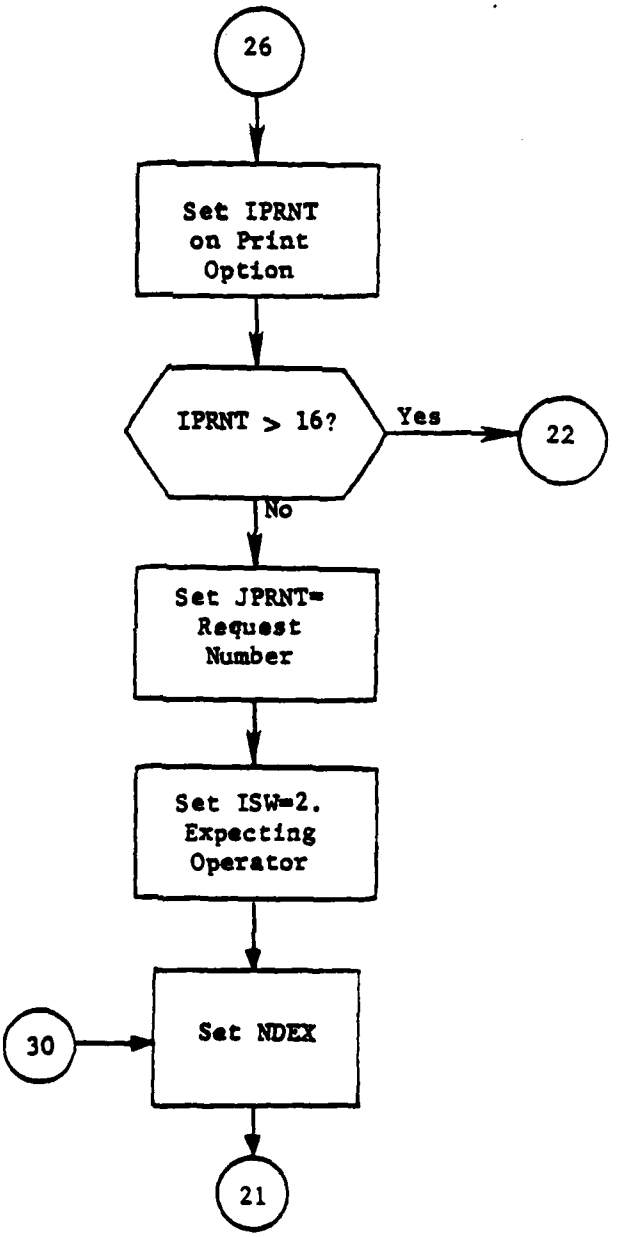


Figure 62. (Part 10 of 13)

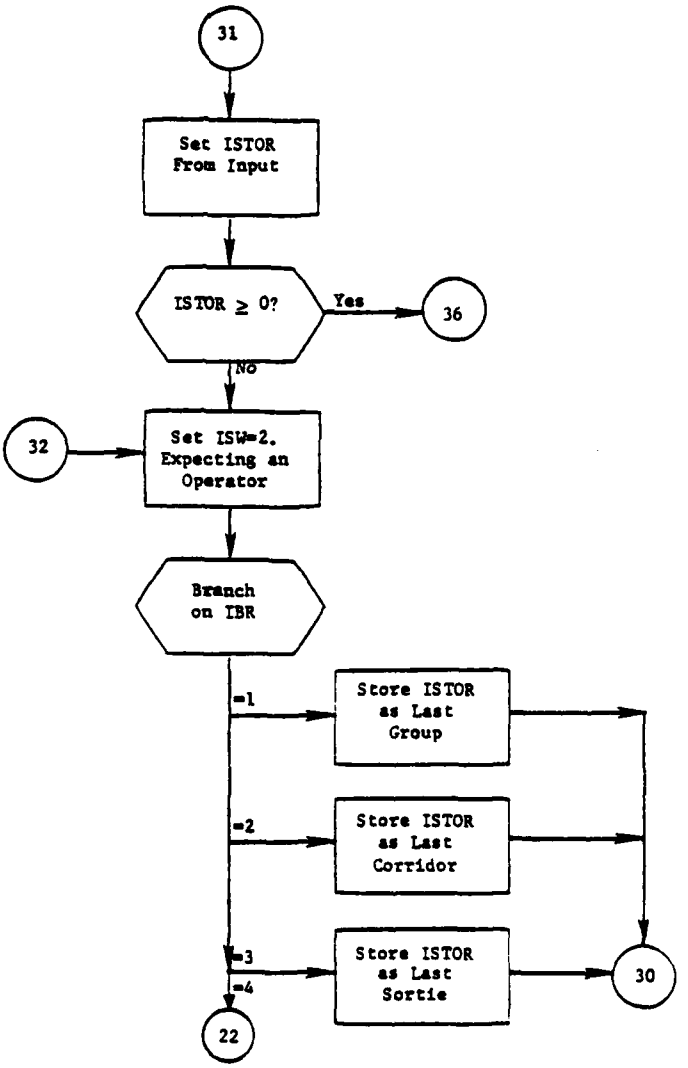


Figure 62. (Part 11 of 13)

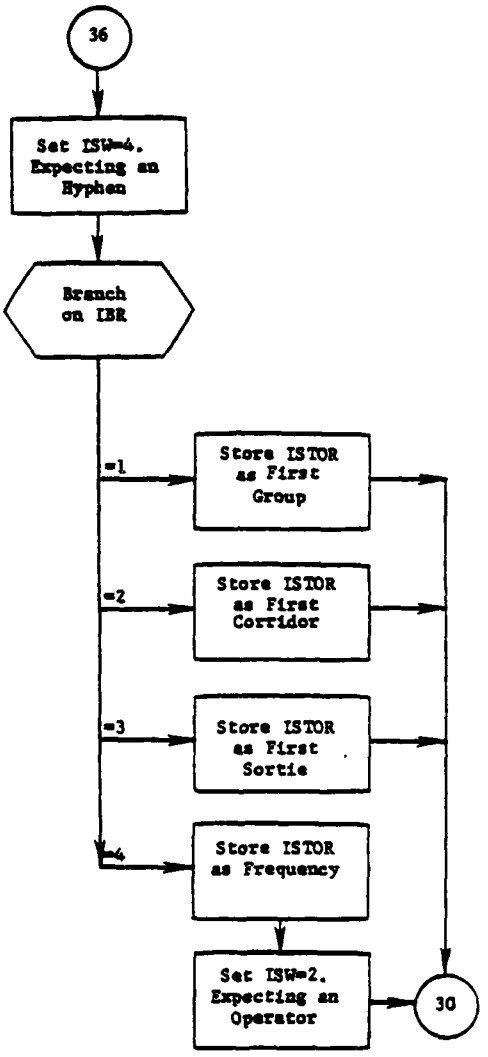


Figure 62. (Part 12 of 13)

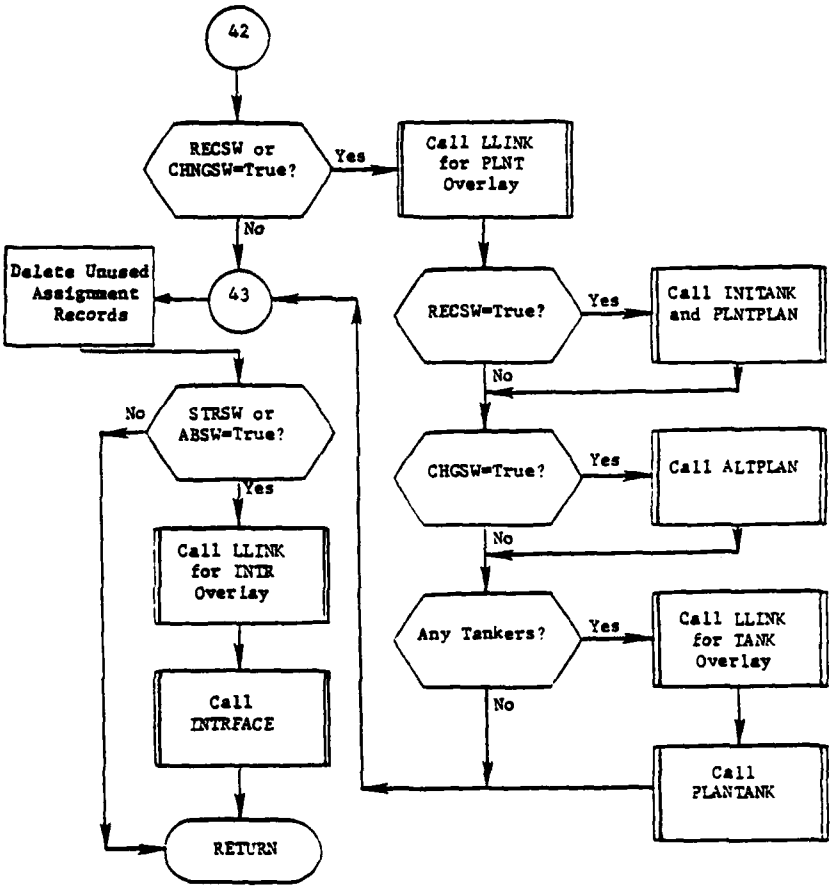


Figure 62. (Part 13 of 13)

4.7.1 Subroutine CLINDATA

PURPOSE: To initialize common /DINDATA/

ENTRY POINTS: CLINDATA

FORMAL PARAMETERS: None

COMMON BLOCKS: DINDATA

SUBROUTINES CALLED: None

CALLED BY: PLAN, PLANTANK

Method:

Each word in common /DINDATA/ is set to zero.

Subroutine CLINDATA is illustrated in figure 63.

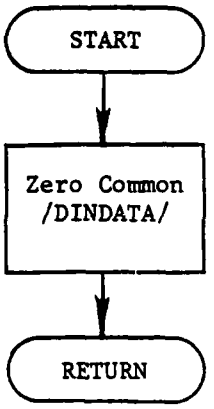


Figure 63. Subroutine CLINDATA

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

CONTENTS OF THIS PAGE INTENTIONALLY DELETED

4.7.3 Subroutine GEOGET

PURPOSE: Collect geography data and store it in appropriate common blocks

ENTRY POINTS: GEOGET

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, CALLSW, CORRCHAR, DPENREF, HAPPEN, RECBAS

SUBROUTINES CALLED: DISTF, HDFND, HEAD, NEXTTT, RETRV

CALLED BY: ALTPLAN, PLNTPLAN

Method:

After checking for a previous call, this subroutine retrieves the penetration corridor header. Then it retrieves each penetration corridor. For each corridor it stores information in the /CORRCHAR/ block. Further, it retrieves the doglegs of each corridor which provide data for both the /CORRCHAR/ and /HAPPEN/ blocks (see section 4.8.13, subroutine PLAN).

Next the header for depenetration corridors is retrieved. Each depenetration corridor is retrieved along with its doglegs and the data stored in blocks /DPENREF/ and /HAPPEN/. Recovery bases linked to the corridors are also retrieved and the associated data store in block /RECBAS/.

The header for recovery bases is now retrieved and the latitude and longitude of each base is saved in block /RECBAS/. Finally all refuel points are stored in block /DPENREF/.

Subroutine GEOGET is illustrated in figure 65.

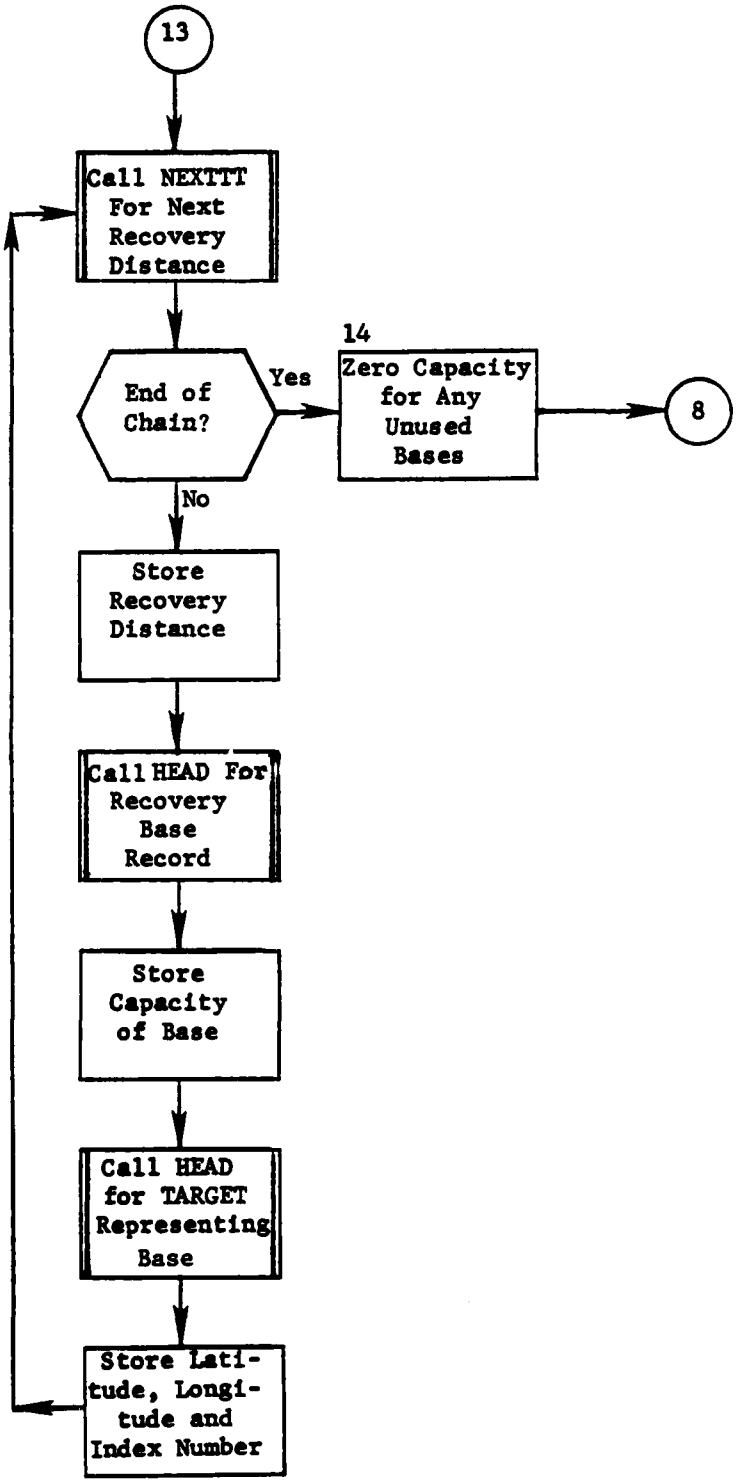


Figure 65. (Part 5 of 6)

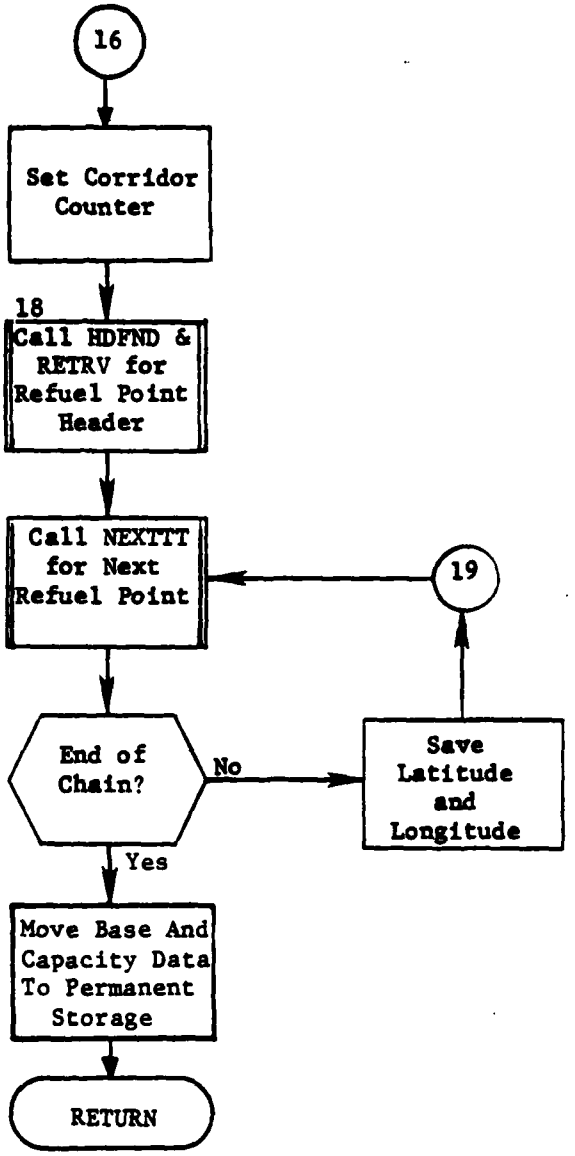


Figure 65. (Part 6 of 6)

4.7.4 Subroutine SNAPCON

PURPOSE: To control the activation of optional prints.

ENTRY POINTS: SNAPCON

FORMAL PARAMETERS: None

COMMON BLOCKS: IFTPRNT, *OUTSRT*, *PRNCON*, *SNAPON*

SUBROUTINES CALLED: None

CALLED BY: ALTPLAN, PLNTPLAN, PLANTANK

Method:

This subroutine with SNAPIT and SNAPOUT provides the capability for optional printing. SNAPCON uses the ONPRINTS clause information to control the activation of prints during processing. SNAPIT is called wherever an optional print is to be issued; SNAPIT, in turn, calls SNAPOUT to do the actual printing.

For each input sortie, SNAPCON checks the print request list with a particular value of a print control parameter, say group, to determine which prints are to be activated. Let x be the value of the print control parameter; e.g., the group number on the current input record. Suppose that a = the starting group and b = the finishing group as specified on the print request card. Then x is checked to determine whether it is in the interval a to b. Either a or b, or both, may be blank or zero on the control card. Table 8 lists the possible values of a and b, and the value that x should assume if the print is to be active in each of these cases. Let a_m be the minimum value x can have and b_m be its maximum value. Then the following single text of x suffices to determine if x is such that the print is active:

$$a' + a_m L(a') \leq x \leq b' + a'L(b') + b_m L(a')$$

where L(x) = 1 if x = 0 and is 0 otherwise.

For each print number (1 to 15) which is to be active for the current plan, SNAPCON sets the corresponding element(s) in the NAP array to 3.

Subroutine SNAPCON is illustrated in figure 66.

Table 8. Possible Values of a and b

a	b	VALUE OF x FOR ACTIVE PRINT
0	0	any value
a'	0	$x = a'$
0	b'	any value
a'	b'	$a' \leq x \leq b'$

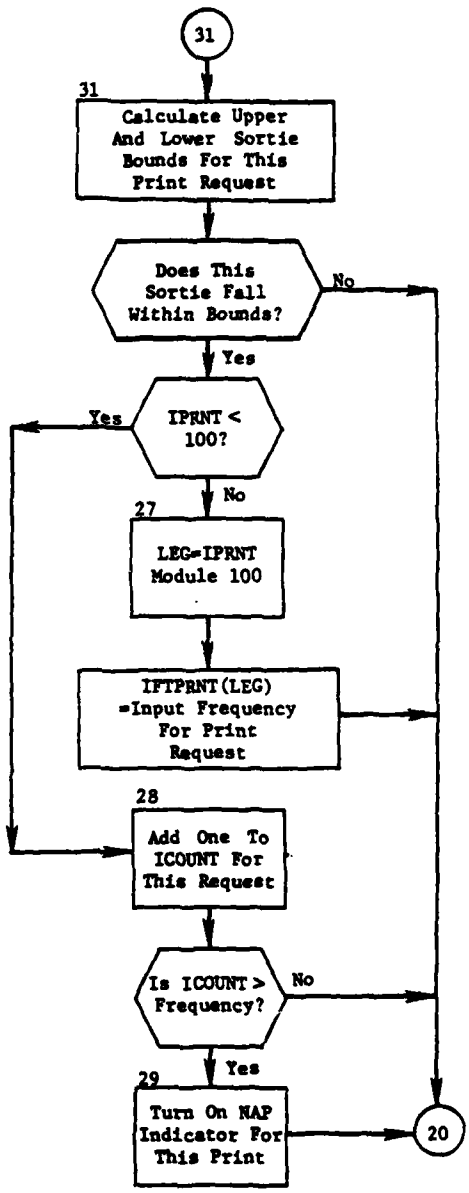


Figure 66. (Part 3 of 3)

4.7.5 Subroutine WEPDATA

PURPOSE: To collect weapon associated data and store it in appropriate common blocks

ENTRY POINTS: WEPDATA

FORMAL PARAMETERS: None

COMMON BLOCKS: C10, C15, C30, CALLSW, GRPSTF, PAYSTF, TYPSTF, WHDSTF

SUBROUTINES CALLED: ABORT, HDFND, HEAD, NEXTTT, RETRV

CALLED BY: ALTPLAN, INTRFACE, PLNTPLAN

Method:

After checking for a previous call, this subroutine retrieves each of the warhead headers (CLASS = BOMB, ASM, RV, MRV and MIRV) and all entries beneath each header. As each entry is retrieved its data, (YIELD and FFRAC) are stored in block /WHDSTF/, additional data is stored when CLASS = ASM. The reference code is also saved in array LREFWHD.

Next the payload table header is retrieved and each payload table is examined. If it is different from all previously stored payload tables, it is stored in block /PAYSTF/. Otherwise a pointer to the previous table is held in LPAYN.

The headers for missile weapons and the bomber weapons are now retrieved and each type entry beneath them accessed. The data from each type entry is stored in block /TYPSTF/ and its reference code saved in LREFTYP.

Finally, the weapon group header is obtained and each weapon group record is retrieved. Group data is saved in block /GRPSTF/. In addition, the LREFTYP and LREFPAY arrays are used to set type and payload indexes respectively.

Subroutine WEPDATA is illustrated in figure 67.

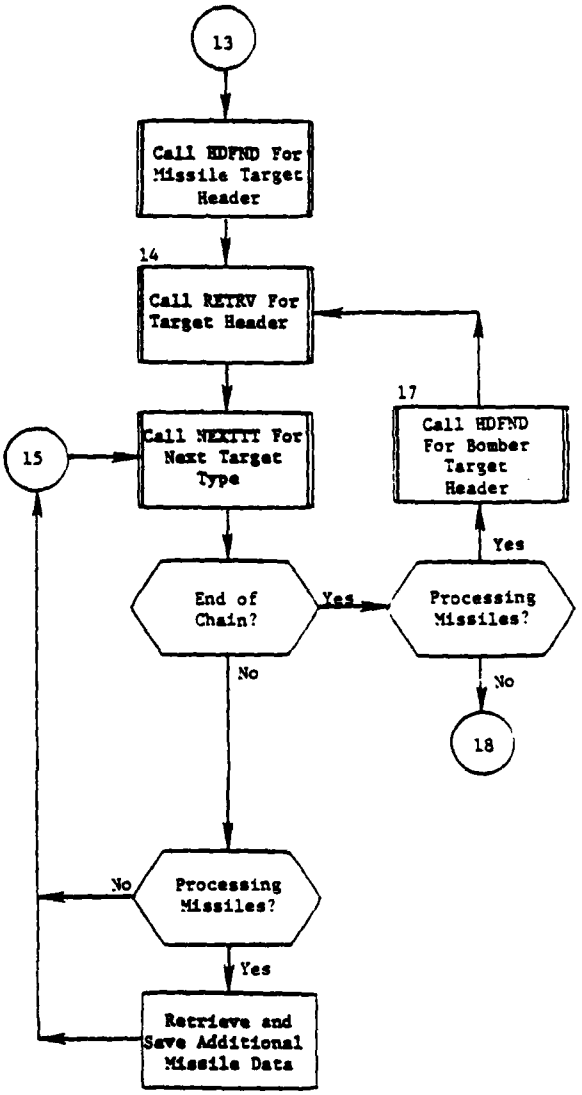


Figure 67. (Part 5 of 7)

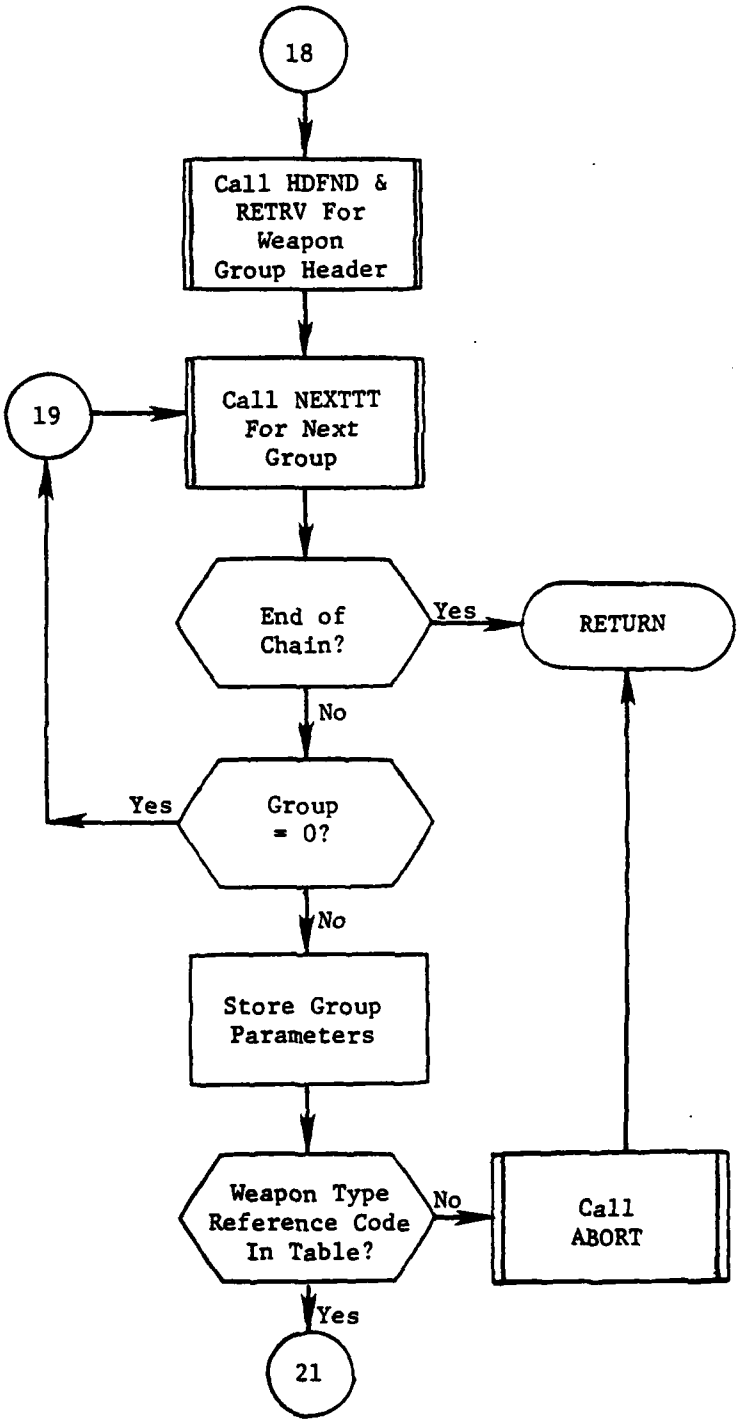


Figure 67. (Part 6 of 7)

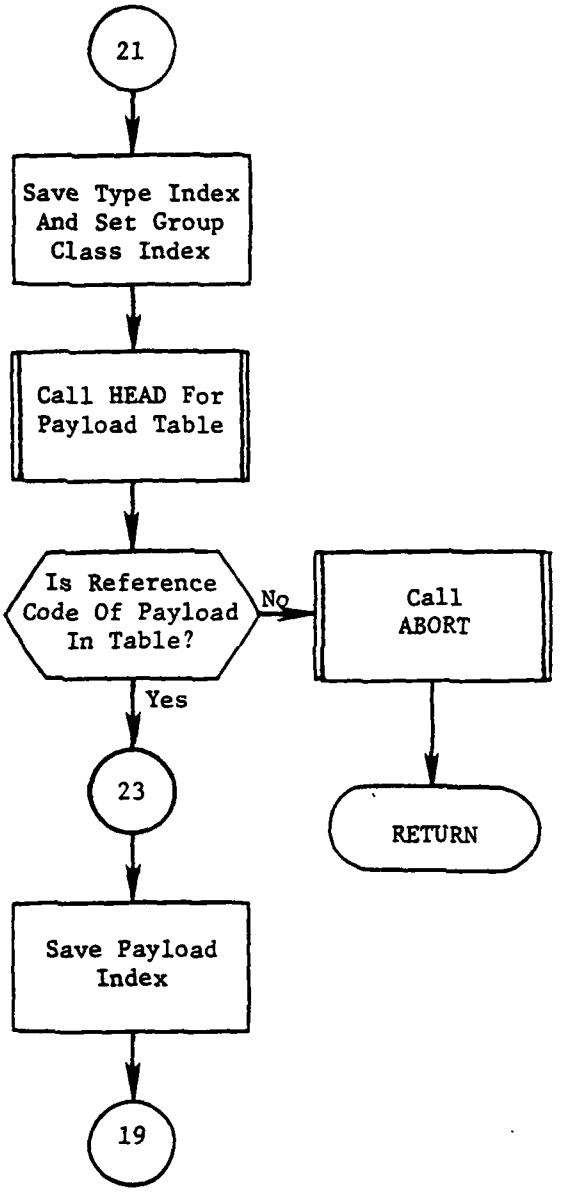


Figure 67. (Part 7 of 7)

CONTENTS OF PAGES 332~334 INTENTIONALLY DELETED

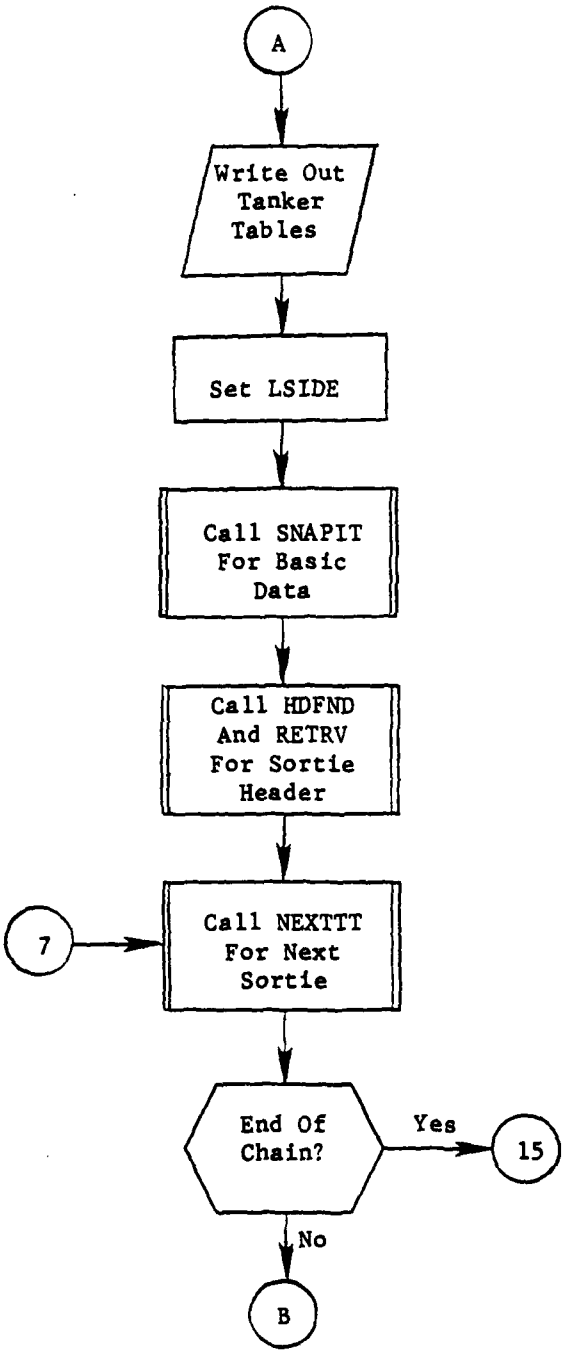


Figure 69. (Part 2 of 7)

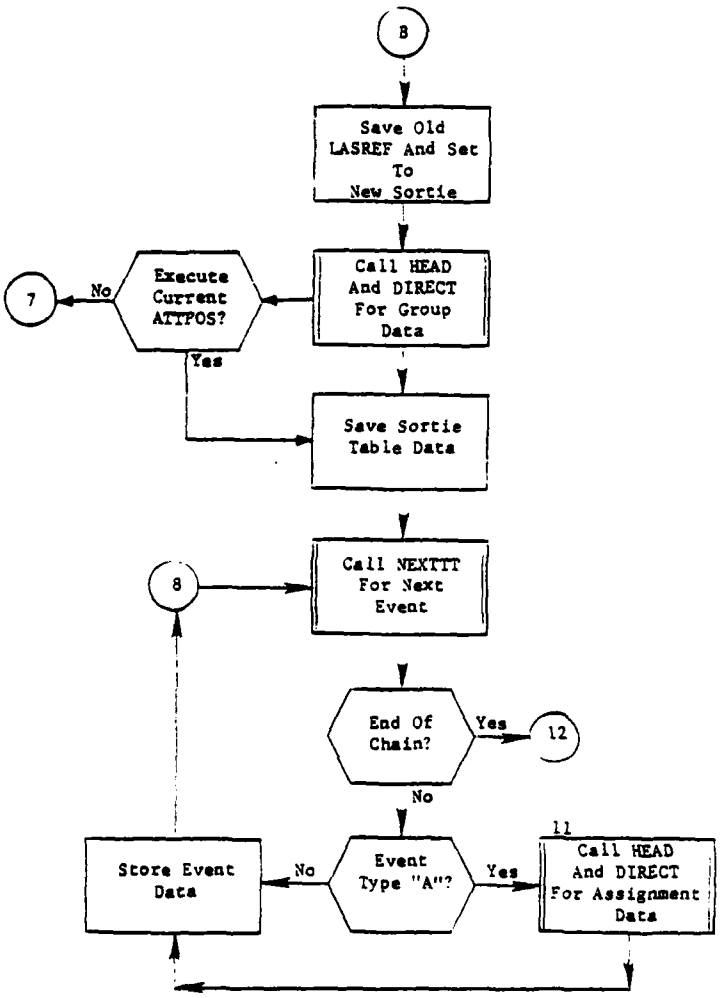


Figure 69. (Part 3 of 7)

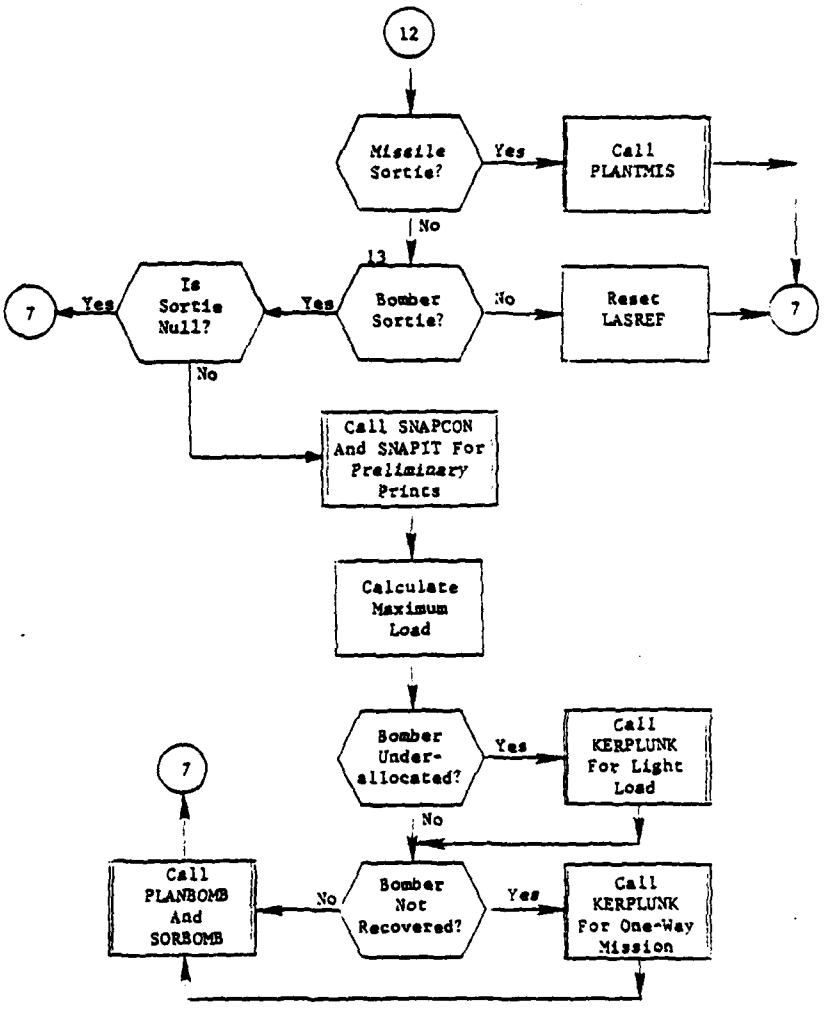


Figure 69. (Part 4 of 7)

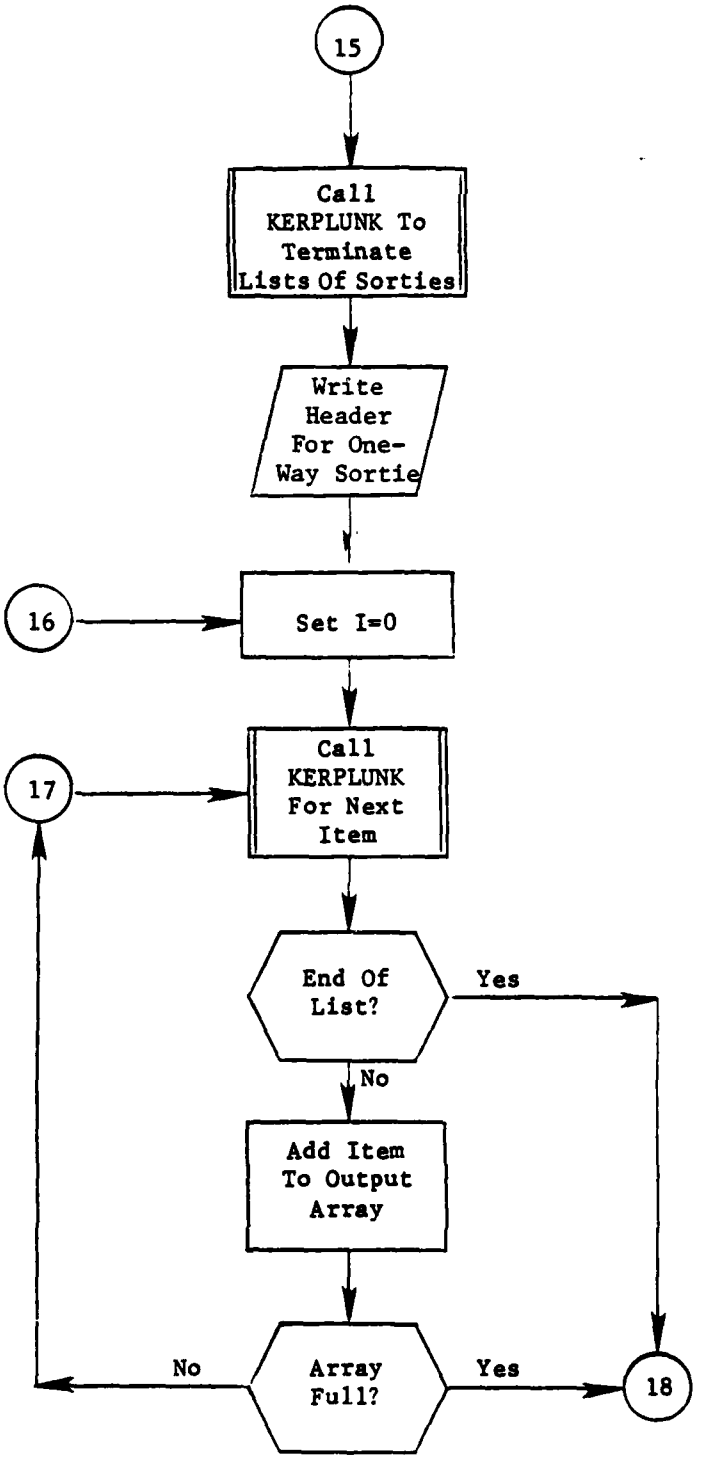


Figure 69. (Part 5 of 7)

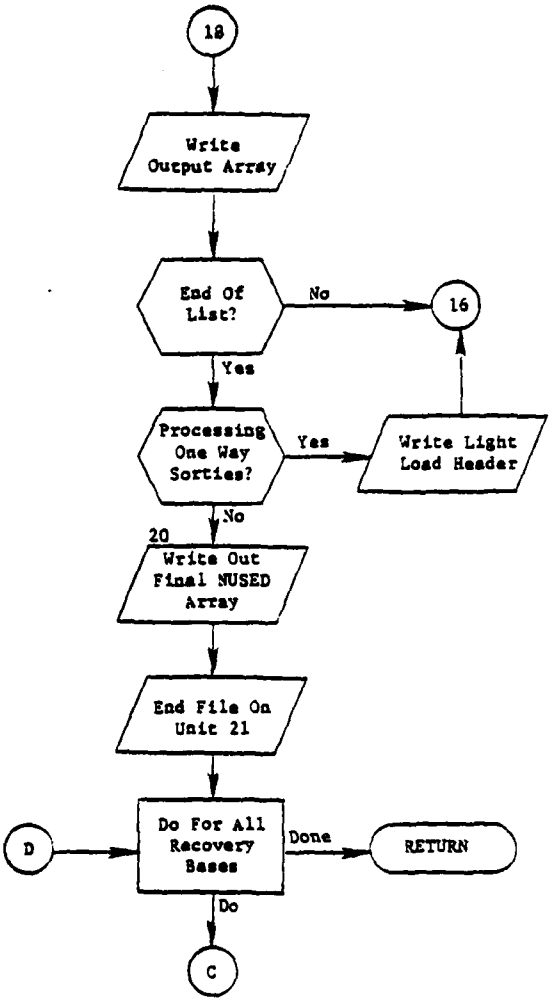


Figure 69. (Part 6 of 7)

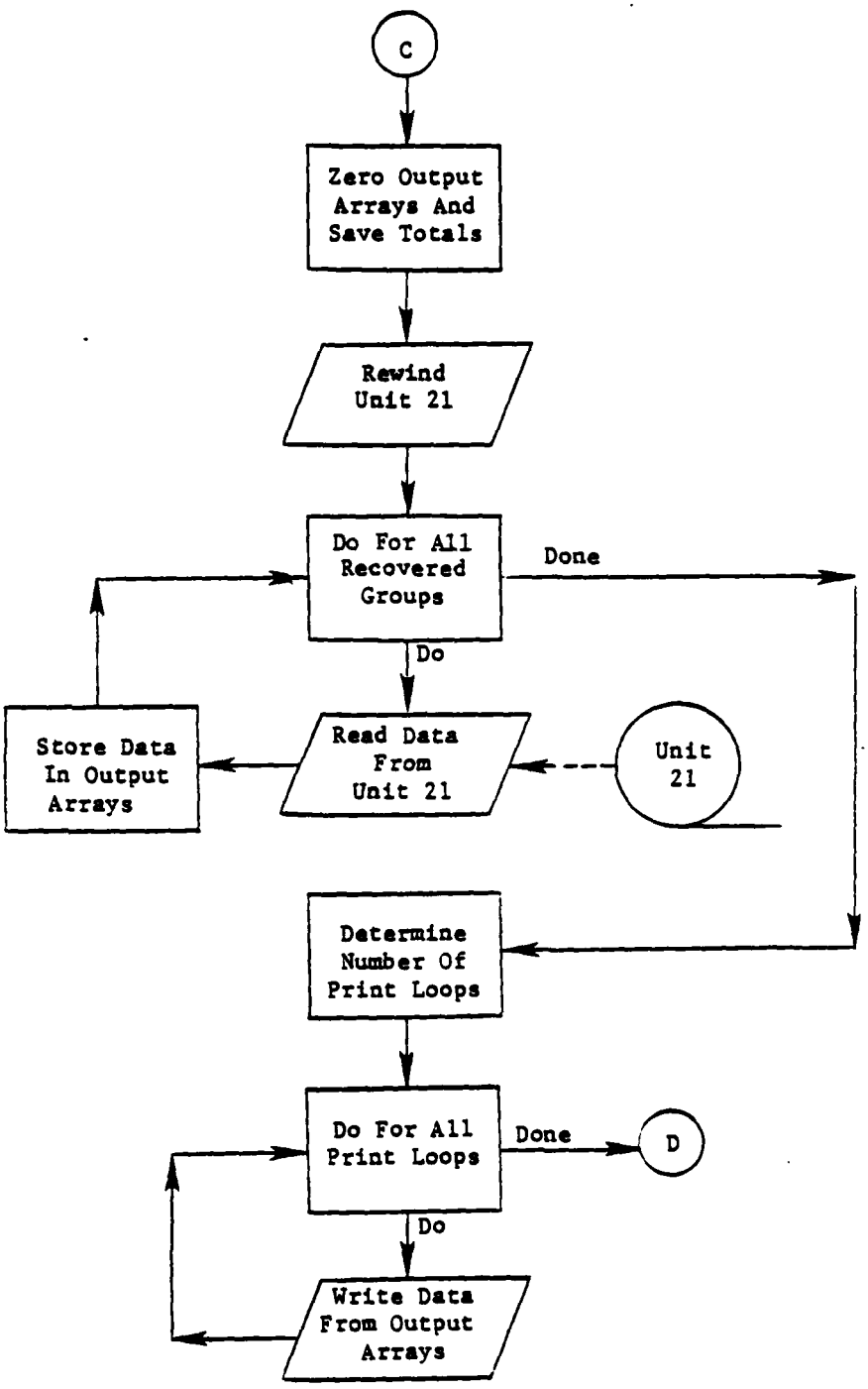


Figure 69. (Part 7 of 7)

4.8.1 Subroutine ALTPLAN*

PURPOSE: Driver for sortie change function.

ENTRY POINTS: ALTPLAN

FORMAL PARAMETERS: None

COMMON BLOCKS: ADVRB, C10, C30, CONTR1, CORRC1, CORRCHAR, ERCOM, GRPSTF, LASREF, OOPS, OUTSRT, TSTUFF, DATEOFC

SUBROUTINES CALLED: DATIM, DIRECT, DISTF, DLETE, FINDME, GEOGET, HEAD, INSGET, INSPUT, LNCHDATA, MODFY, NEXTTT, ORDER, PLANBOMB, PLANTMIS, RETRV, SLOG, SNAPCON, SORBOM, STORE, TOFM, WEPDATA, ALTERR

CALLED BY: ENTMOD (PLANOUT)

Method:

User Inputs

The user may alter any sortie developed within the QUICK system. Subroutine ALTPLAN, reads these sortie change requests and updates the data base accordingly. Three adverbs are recognized and are:

'CCARD' -- indicates change strike
'ICARD' -- indicates insert strike
'ACARD' -- indicates add sortie

The CCARD Clause. The general form of the CCARD (see UM 9-77, Volume IV for complete discussion) is:

CCARD sortie number, desig1, desig2 [, hob, dec, rac, tchange, asm] [* caloff, dlatoff, dlongoff]

The first three parameters specify the action to be performed and must be entered for each adverb. The "sortie number" indicates which sortie is to be changed. Various modes of entries for the target DESIG's are:

- o "desig1" will be dropped when "desig2" is blank (that is, a comma appears in lieu of a target DESIG)
- o Strikes are replaced when both "desig1" and "desig2" are non-blank and not equal. "desig1" will be replaced by "desig2" (and if a complex, it must be the representative target)

* Second subroutine of overlay PLNT. This overlay performs both sortie completion and sortie change functions. The sortie change function driver is ALTPLAN.

- When "desig1" equals "desig2", elements of the strike are changed. This allows a change in down time, height of burst, offset characteristics or depenetration corridor.

Following "desig2" parameters that may be changed are optional. These options come in two collections. In each collection the individual parameters may be omitted but their preceding commas must still appear. The first collection contains the options of changing the height-of-burst, specifying a new depenetration corridor, suppressing recalculation of attrition, and altering the flight time. The second collection permits the definition of target offset (DEGREES, MINUTES, SECONDS, DIRECTION (N,S,E,W) OR ZERO). This collection must be introduced by the asterisk (*) operator. Also, if no options are used from the first collection, default commas are not required. Similar logic applies if the fourth, or third and fourth, or second, third or fourth options are not employed.

The ICARD Clause. The general form of the clause is

ICARD sortie number , [desig1] , desig2 [, hob , dec , rac , tchange] [* caloff , dlatoff , dlongoff]

"desig2" will be inserted after "desig1". If "desig1" is omitted (two commas after the sortie number), "desig2" will become the first target of the sortie. The discussion of optional information for the CCARD clause on new targets applies to "desig2".

The ACARD Clause. This clause is used to add nonMIRV missile sorties. This clause has the general form:

ACARD desig, hob, group, siteind, [isal, tlaun] [* caloff, dlatoff, dlongoff]

Generally, all comments concerning the CCARD clause apply to the ACARD clause. Note, that no sequence number is supplied; PLANOUT will supply the correct value. "group" is the weapon group number containing the launch base. "siteind" is the INDEXNO of the site from the weapon group. "isal" is the salvo number to be associated with the added sortie. If not input, "isal" will default to zero for non-salvoed sorties and to one for salvoed sorties. "tlaun" is the delay time, in hours, to be applied before launch of the sortie. For non-salvoed sorties, the launch time is simply h-hour + "tlaun", assuming no MISTIME/MSLCOR clauses were input. For salvoed missiles, the launch time is h-hour + ("isal" -1) * LCHINT/60. + "tlaun". If simultaneous launches are desired for salvoed sorties, "isal" must be repeated for each round which is to be salvoed, i.e., if SIMLAUNCH is i, the missile salvo number j would be repeated i times in order to have i weapons launch at (j-1)*LCHINT/60. + "tlaun".

Processing within ALTPLAN closely follows user requests and is described below.

General

This subroutine is best followed by reference to figure 70 as much of the logic is involved in data checking of input clauses. After preliminary calls to obtain the necessary data for execution and to FINDME which retrieves desired group, sortie and target records, the subroutine begins to process the input clauses one at a time.

ACARD Processing

If the clause encountered is an ACARD clause, the subroutine checks the clause for errors. In the process of this check the desired target and group records are retrieved by FINDME. The last nontanker sortie is now retrieved. The range to the target is checked for system limits and the time of flight, salvo, and launch time calculated. The values for the new ASSIGN (record type 70) record are stored in /C30/ and STORE is called. Finally, the values for the SRTEVA (record type 50) record are stored in /C30/ and STORE is called. When all record storage is complete, PLANTMIS is called to complete the sortie. Then all sorties which occur after the new sortie (i.e., tanker sorties) ar retrieved and modified with their sortie numbers incremented.

CCARD and ICARD Processing

If the clause encountered in processing is a CCARD or ICARD clause, all remaining clauses are checked to see if they involve the same sortie. All such clauses (all those with the same sortie number) are then processed before any clauses involving other sorties are processed.

When the first set of change clauses (CCARD and/or ICARD) for any sortie are processed, the contents of common block /CORRC1/ are calculated. This block contains information on defended areas of penetration corridors for subroutine FLTSORT.

Each applicable clause is error checked in turn. In the process FINDME retrieves the appropriate sortie table (SRTYTB). Furthermore, unless the first input DESIG value is blank, the TARCDE record corresponding to the target identified by the first input DESIG value is retrieved and the first attendent weapon assignment (ASSIGN) from the proper group found. Alternately, the user may have entered a strike number and this number is used to find the desired strike and, therefore, DES1. The remainder of the process depends upon whether the clause was a CCARD or ICARD clause and the values of the two DESIGs input (DES1 and DES2).

In the case of a CCARD the second DESIG(DES2) is checked. If DES2 is blank, the ASSIGN record retrieval above is deleted. If DES1 equals DES2, the subroutine checks on which data items have been modified and then the ASSIGN, SRTEVA and SRTYTB records are modified as required. If DES1 is not equal DES2 and DES2 is a legal target DESIG, the old ASSIGN record is deleted and the process proceeds as in an ICARD below.

For an ICARD (or from a CCARD above), the process is to create a new ASSIGN and SRTEVA record in the proper chain order. This order is established by the value of DES1. If DES1 is blank, the new assignment appears as the first assignment event. In any case the target list record for DES2 is retrieved and the new ASSIGN and SRTEVA records stored as per the input clause.

When all clauses pertaining to a particular sortie have been processed, the subroutine carries out the completion process for that sortie as follows: If the sortie is a missile sortie *PLANTMIS* is called. In the case of a bomber sortie the process is more complex. The sortie is completely read into the /OUTSRT/ block. Then, unless specified otherwise by the user, the FLTSORT subroutine is called to recalculate flight time parameters. The PLANBOMB is called to complete the sorties and SORBOMB to update the integrated data base.

At the end of all processing, ALTPLAN produces requested sortie prints if RECALC mode is not active.

Subroutine ALTPLAN is illustrated in figure 70.

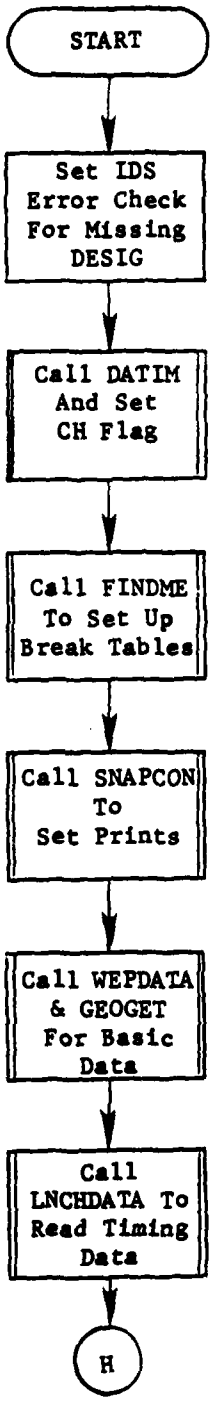


Figure 70. Subroutine ALTPLAN (Part 1 of 35)

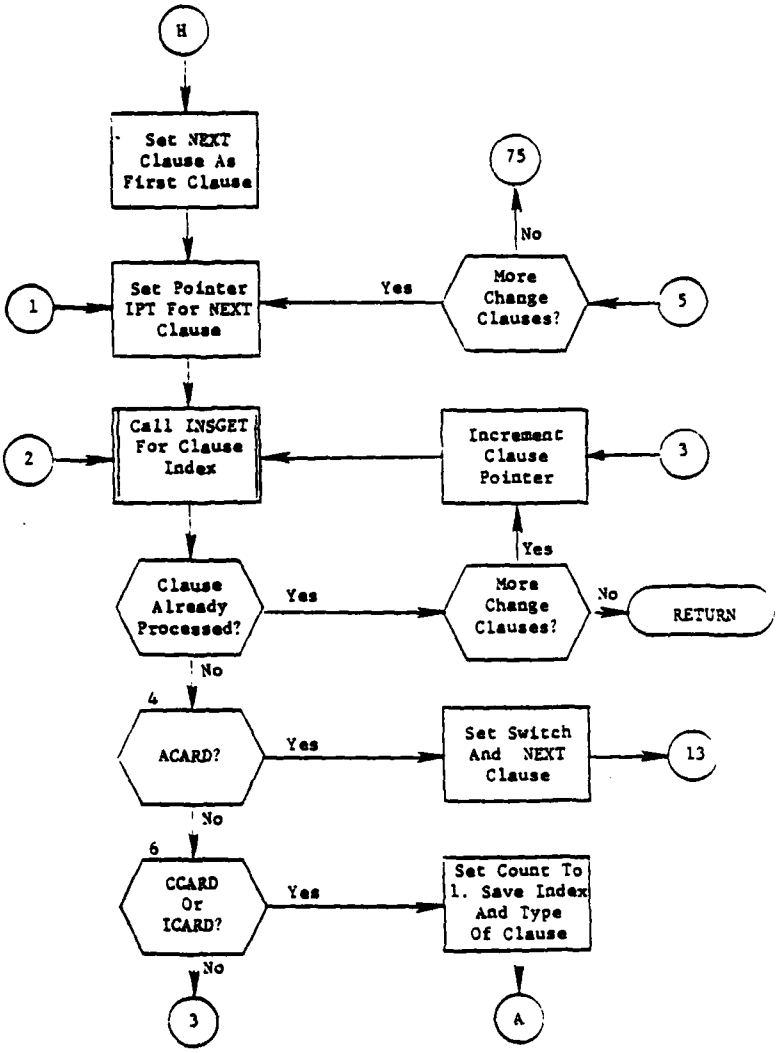


Figure 70. (Part 2 of 35)

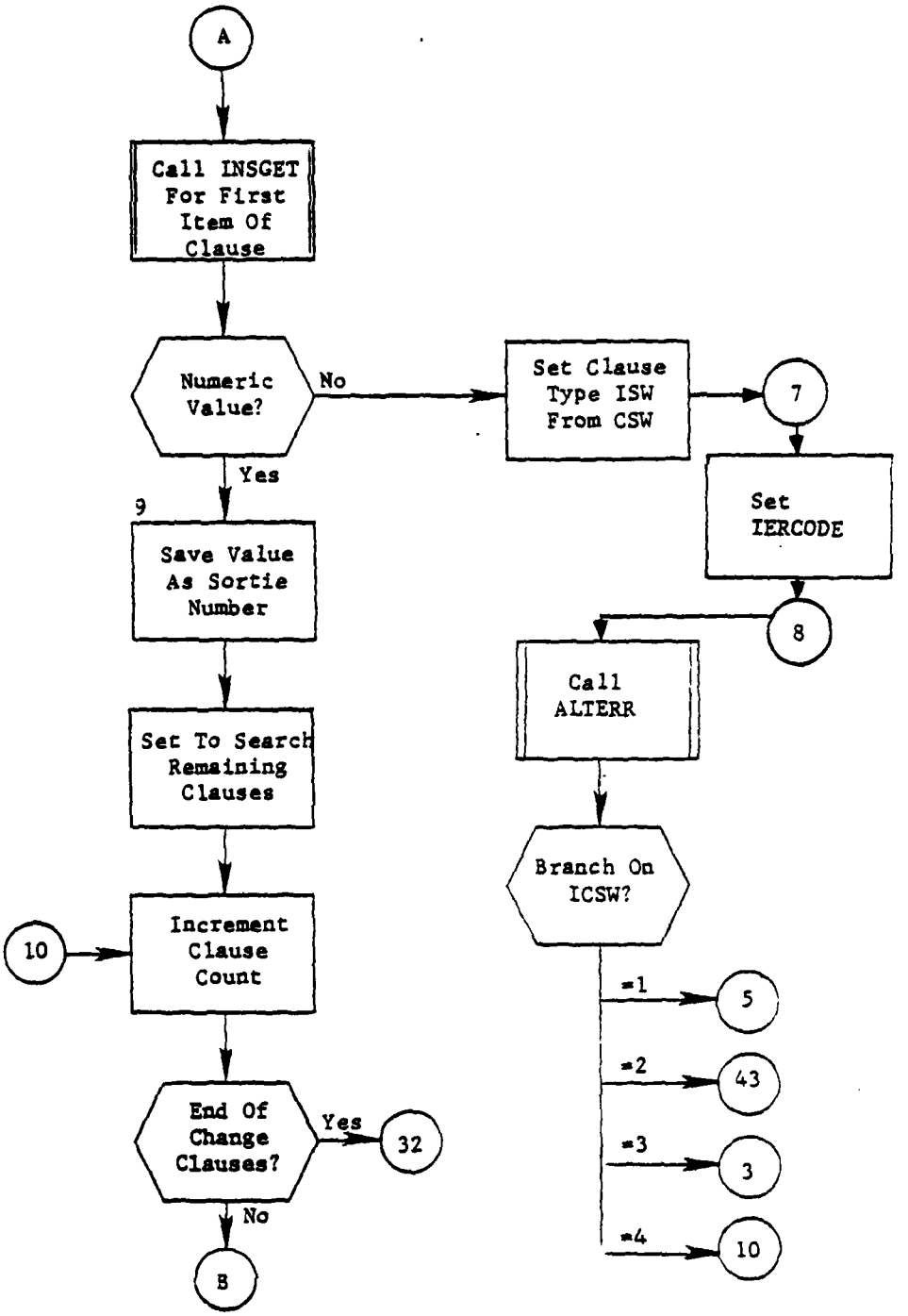


Figure 70. (Part 3 of 35)

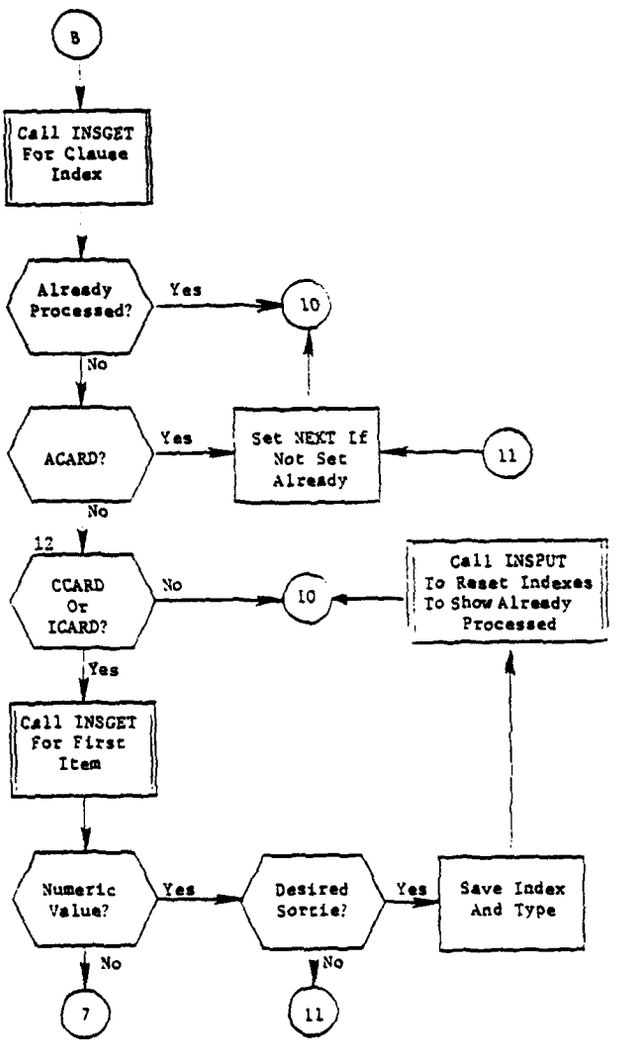


Figure 70. (Part 4 of 35)

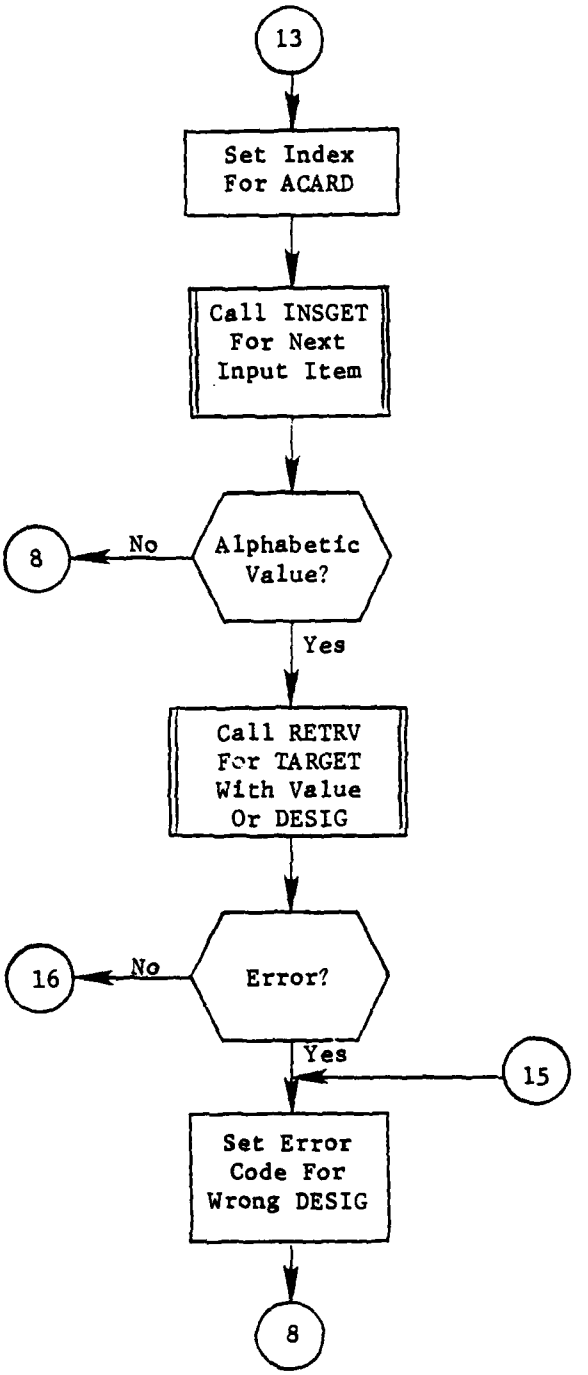


Figure 70. (Part 5 of 35)

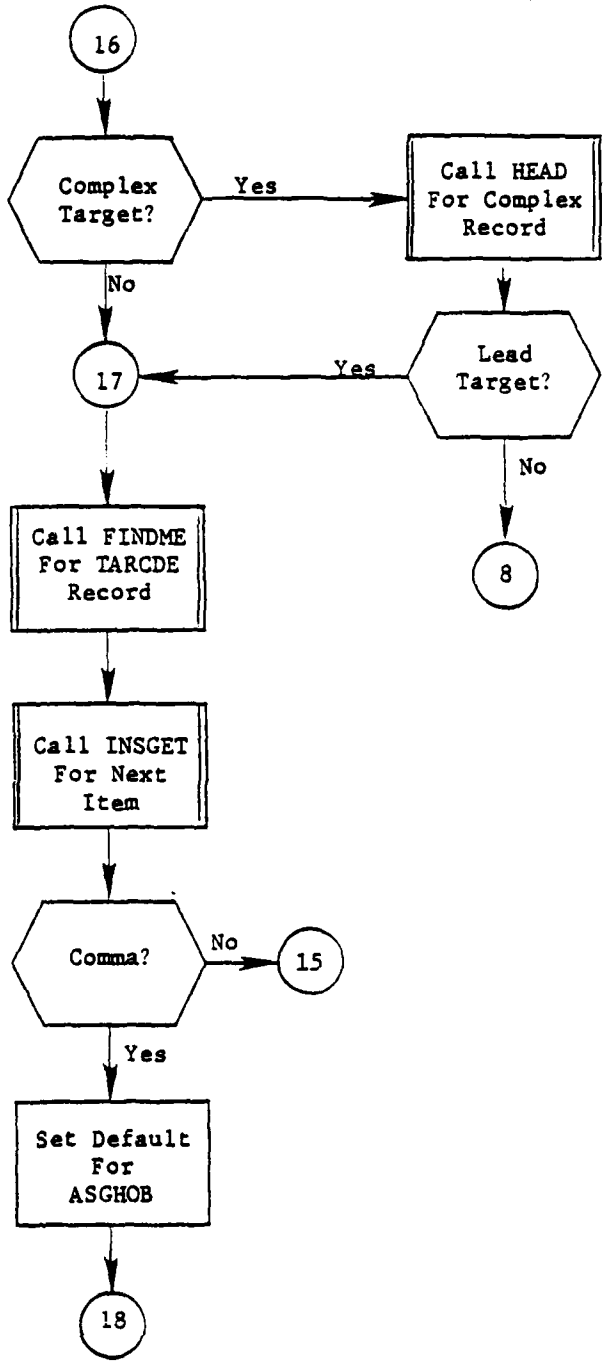


Figure 70. (Part 6 of 35)

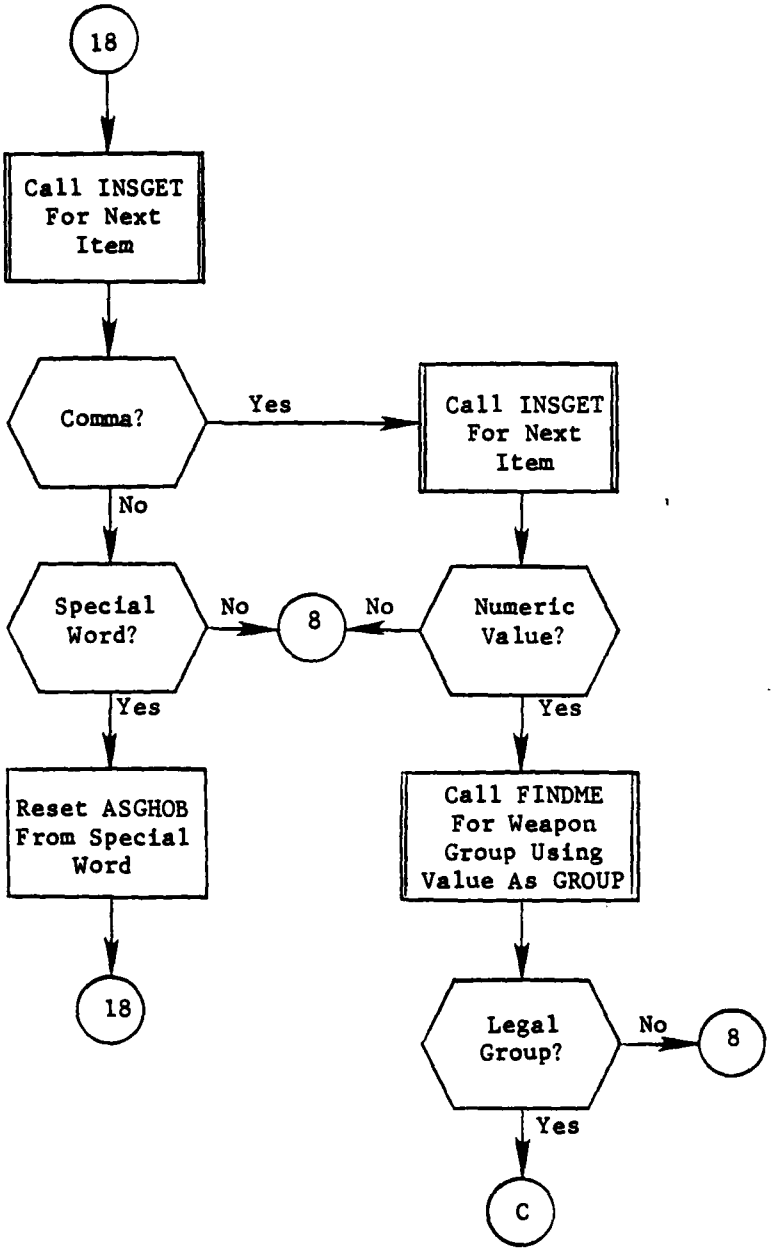


Figure 70. (Part 7 of 35)

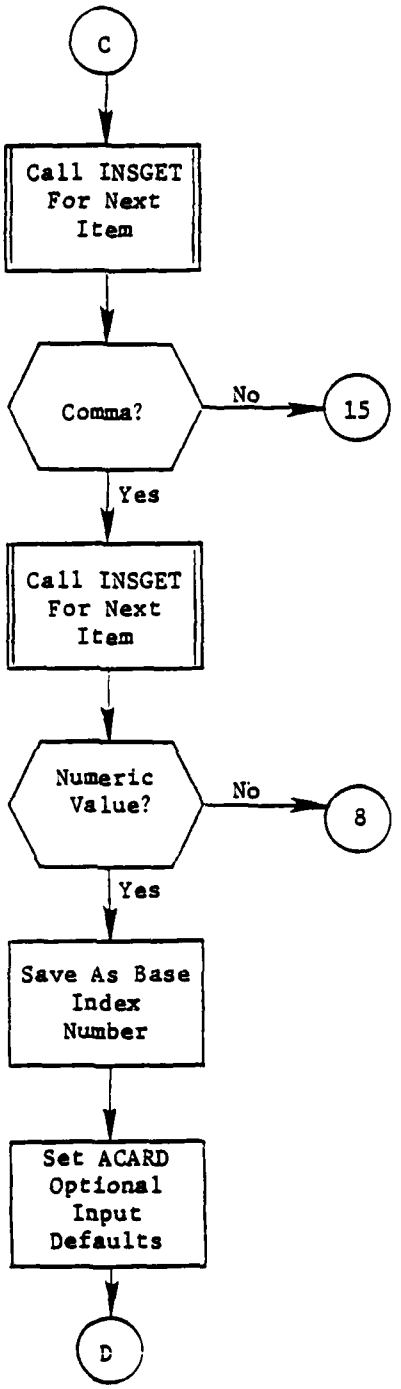


Figure 70. (Part 8 of 35)

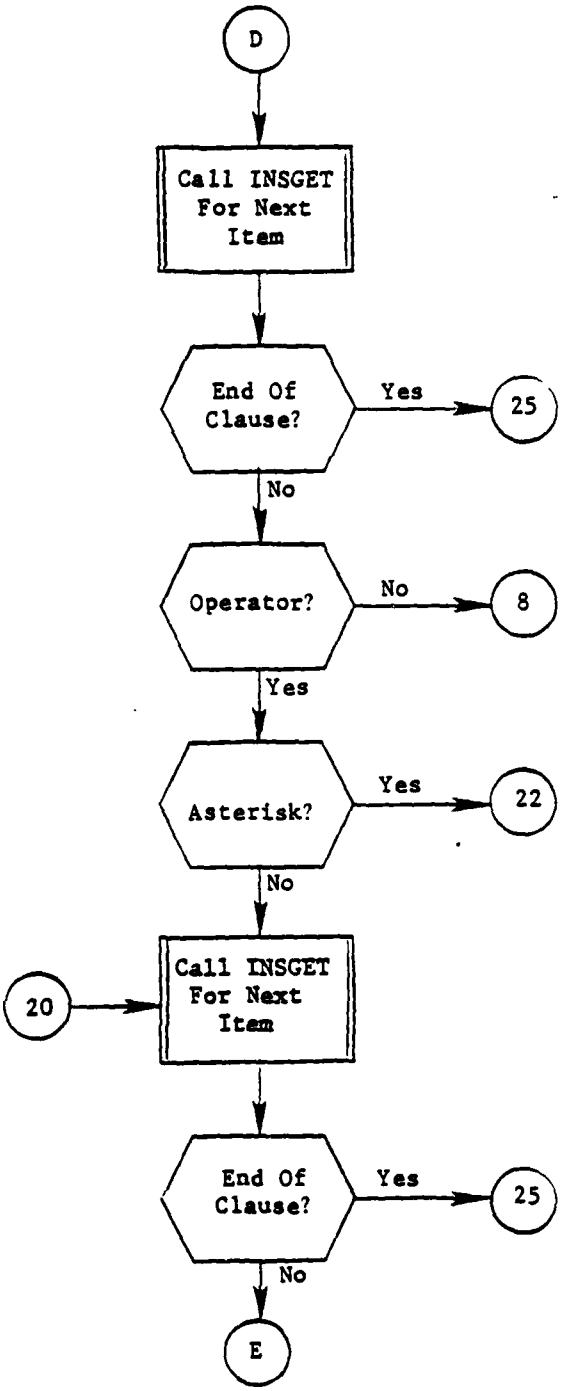


Figure 70. (Part 9 of 35)

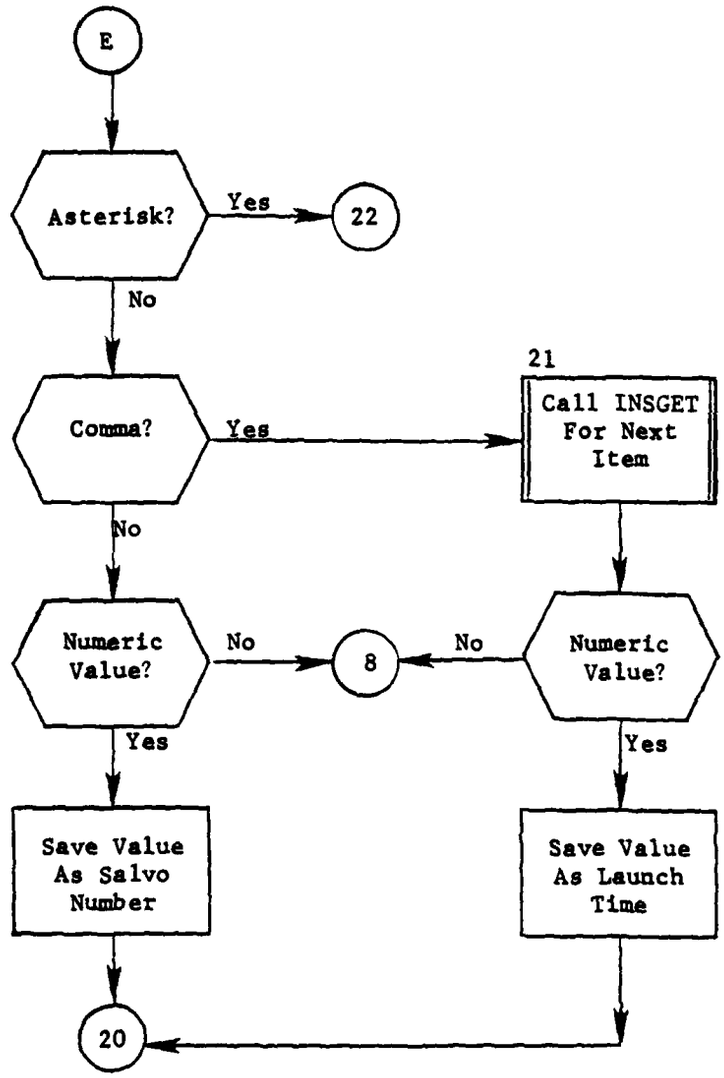


Figure 70. (Part 10 of 35)

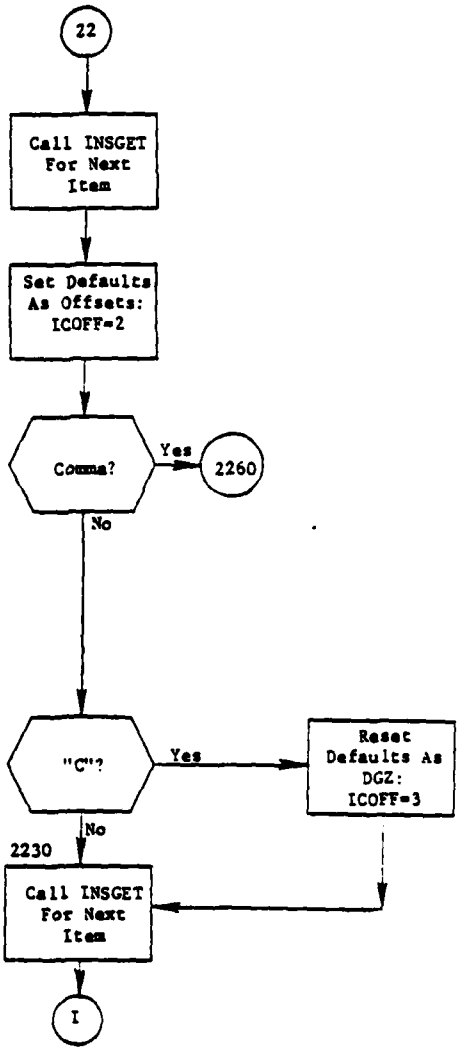


Figure 70. (Part 11 of 35)

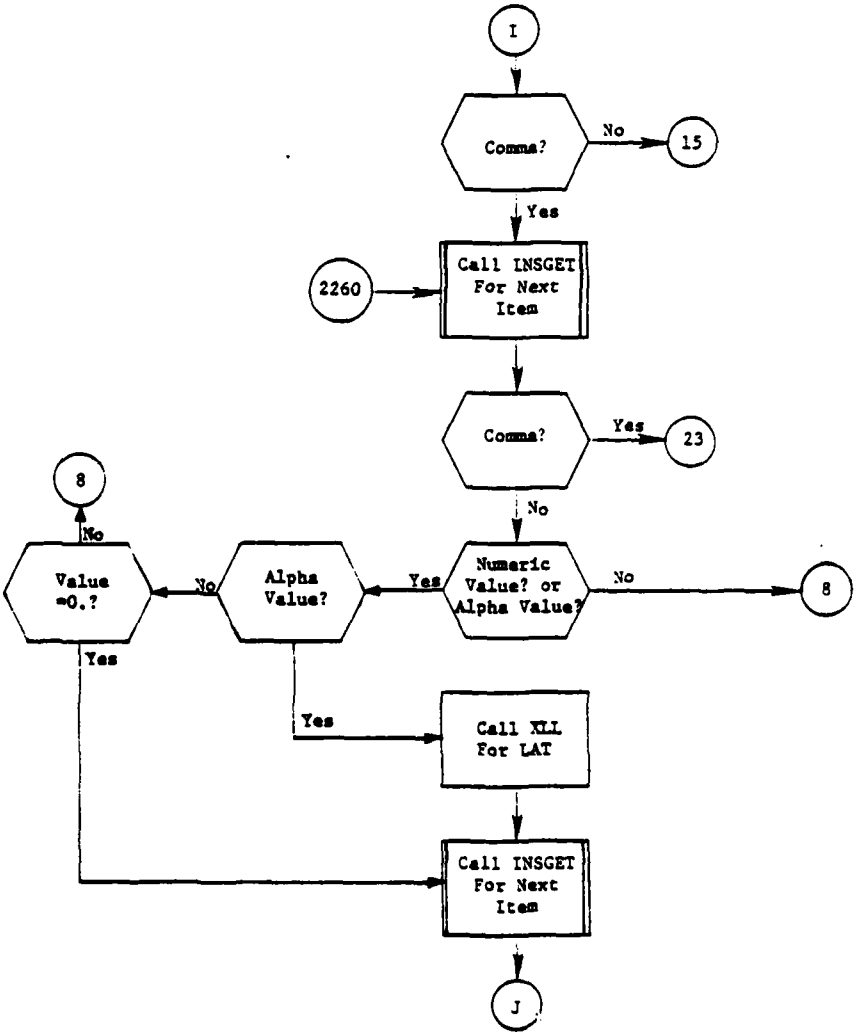


Figure 70. (Part 12 of 35)

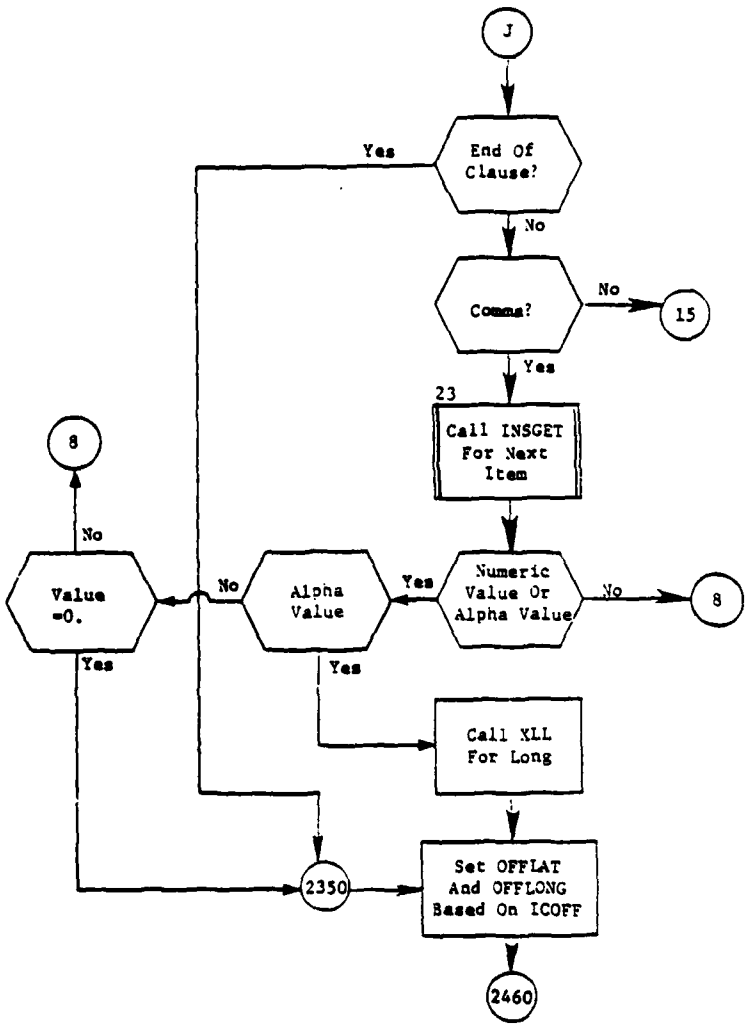


Figure 70. (Part 13 of 35)

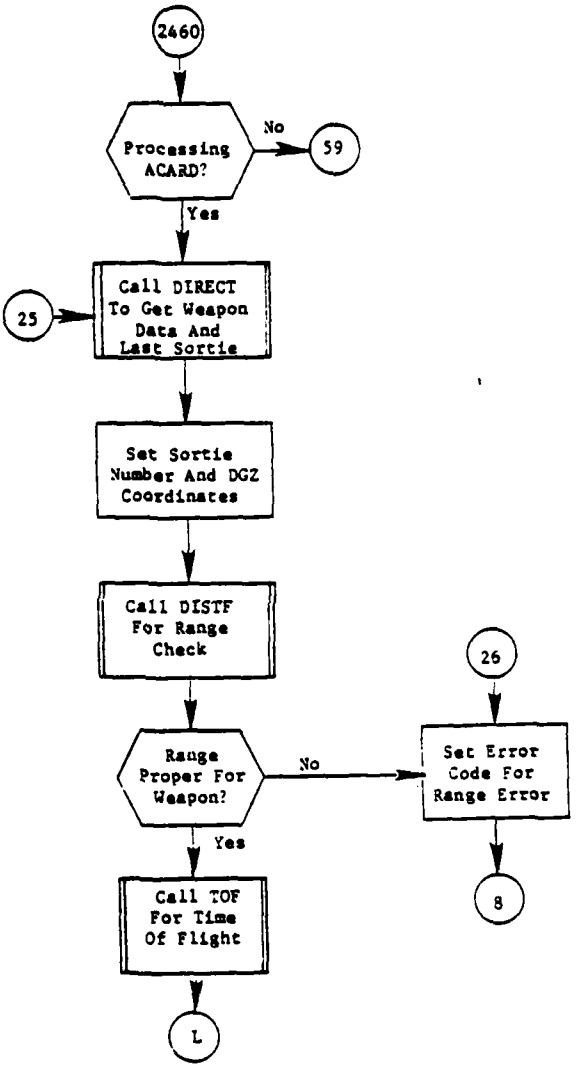


Figure 70. (Part 14 of 35)

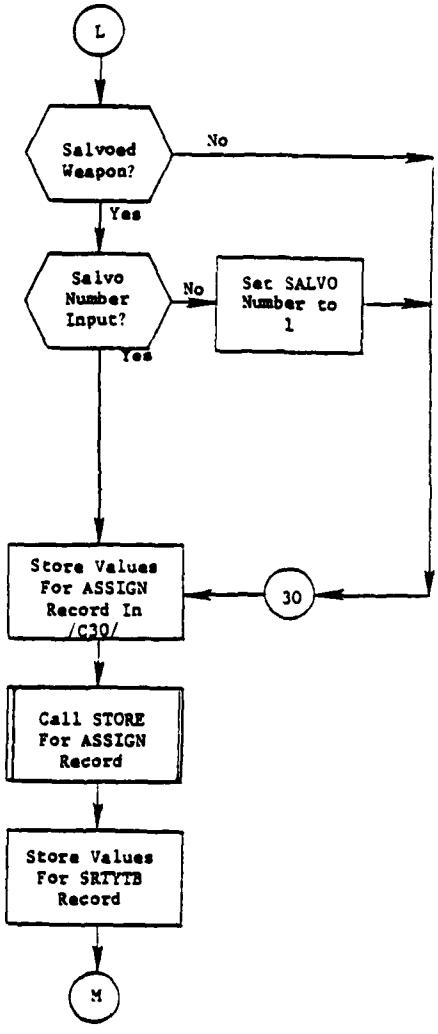


Figure 70. (Part 15 of 35)

Figure 70. (Part 16 of 35)

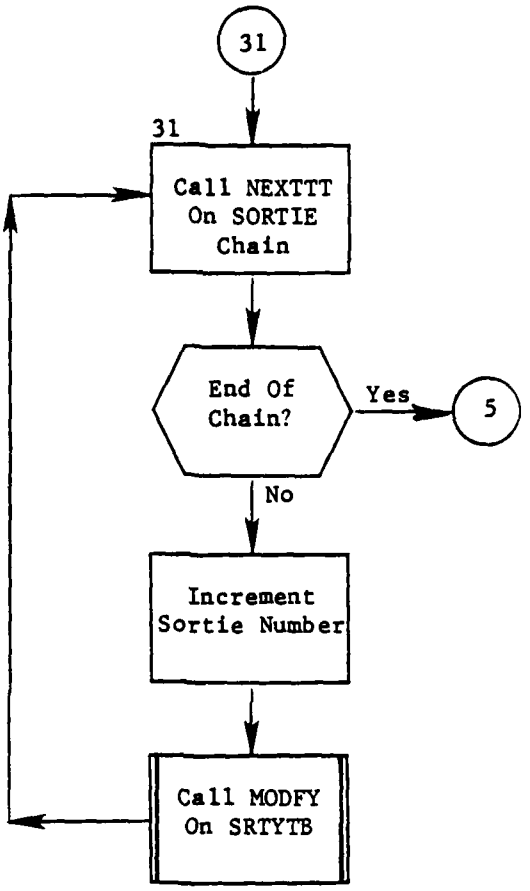


Figure 70. (Part 17 of 35)

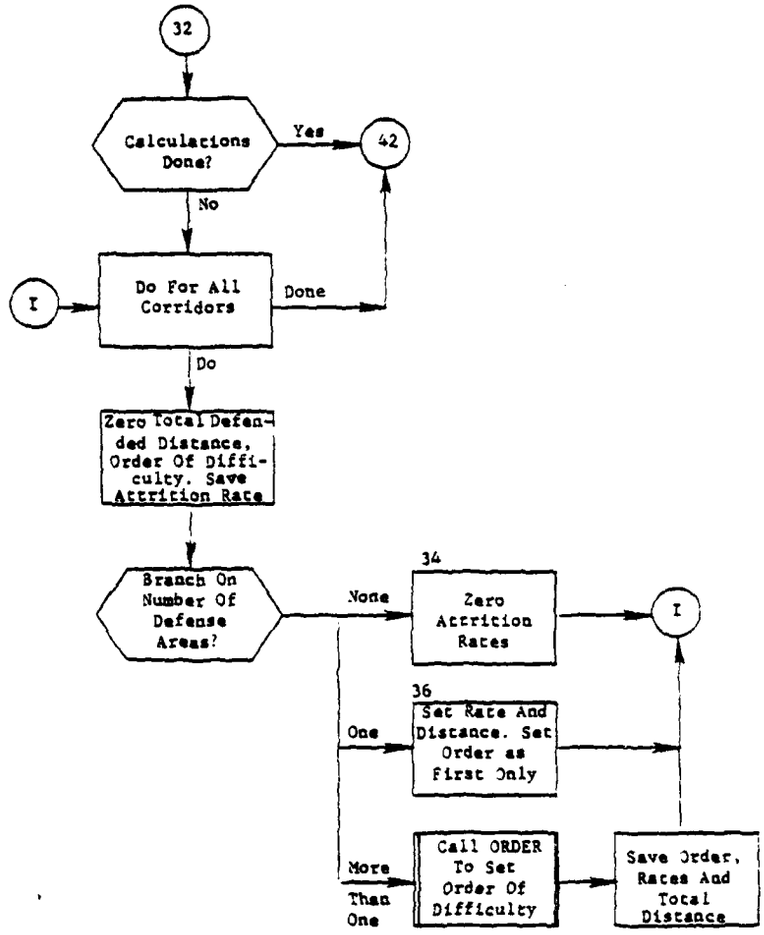


Figure 70. (Part 18 of 35)

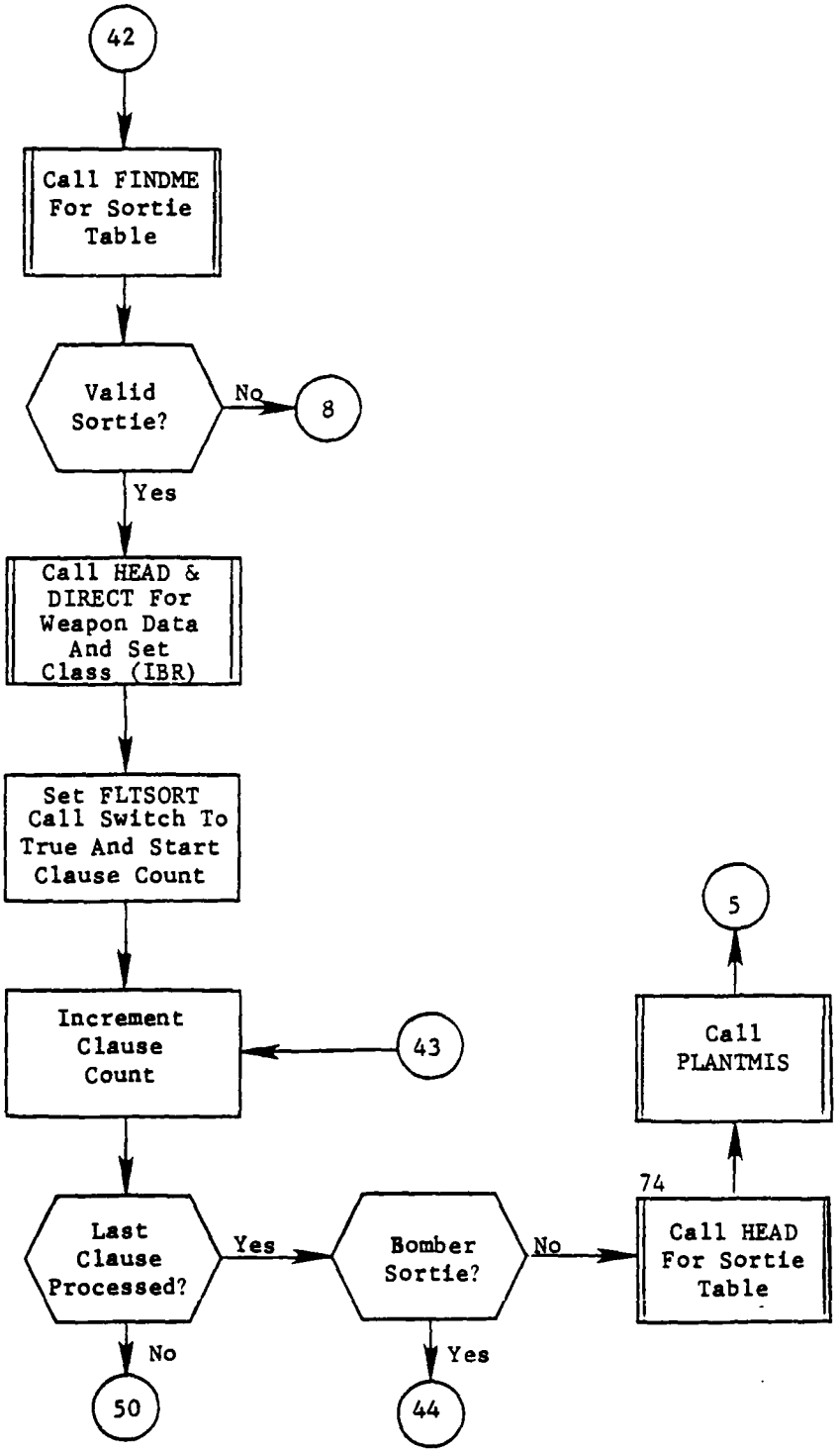


Figure 70. (Part 19 of 35)

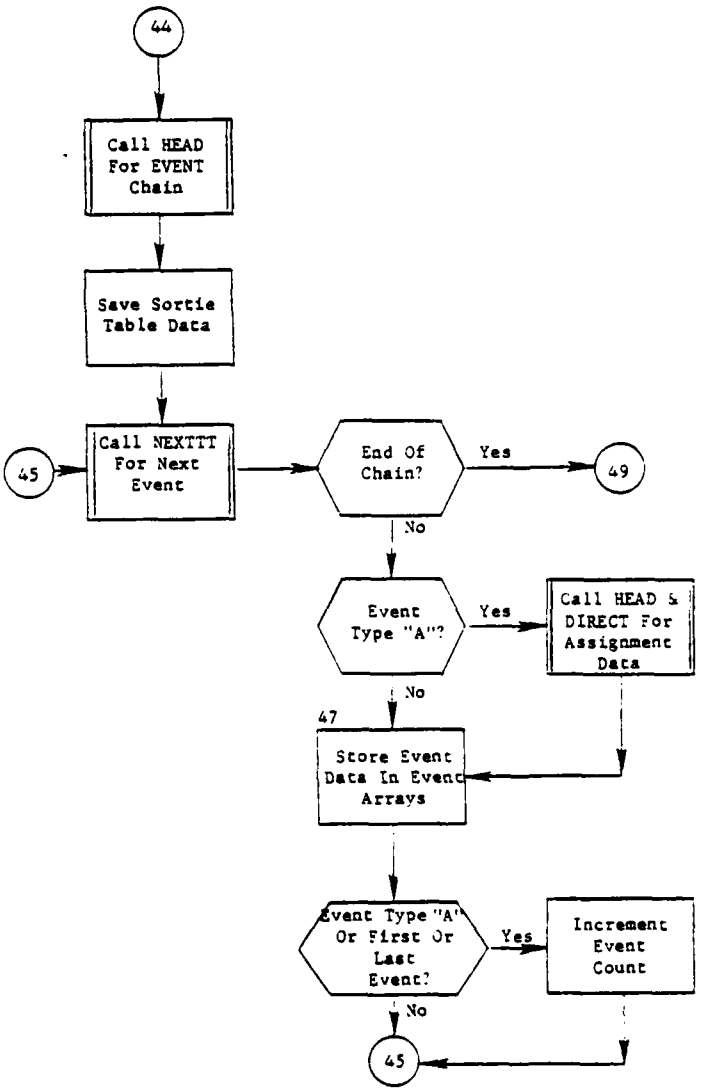


Figure 70. (Part 20 of 35)

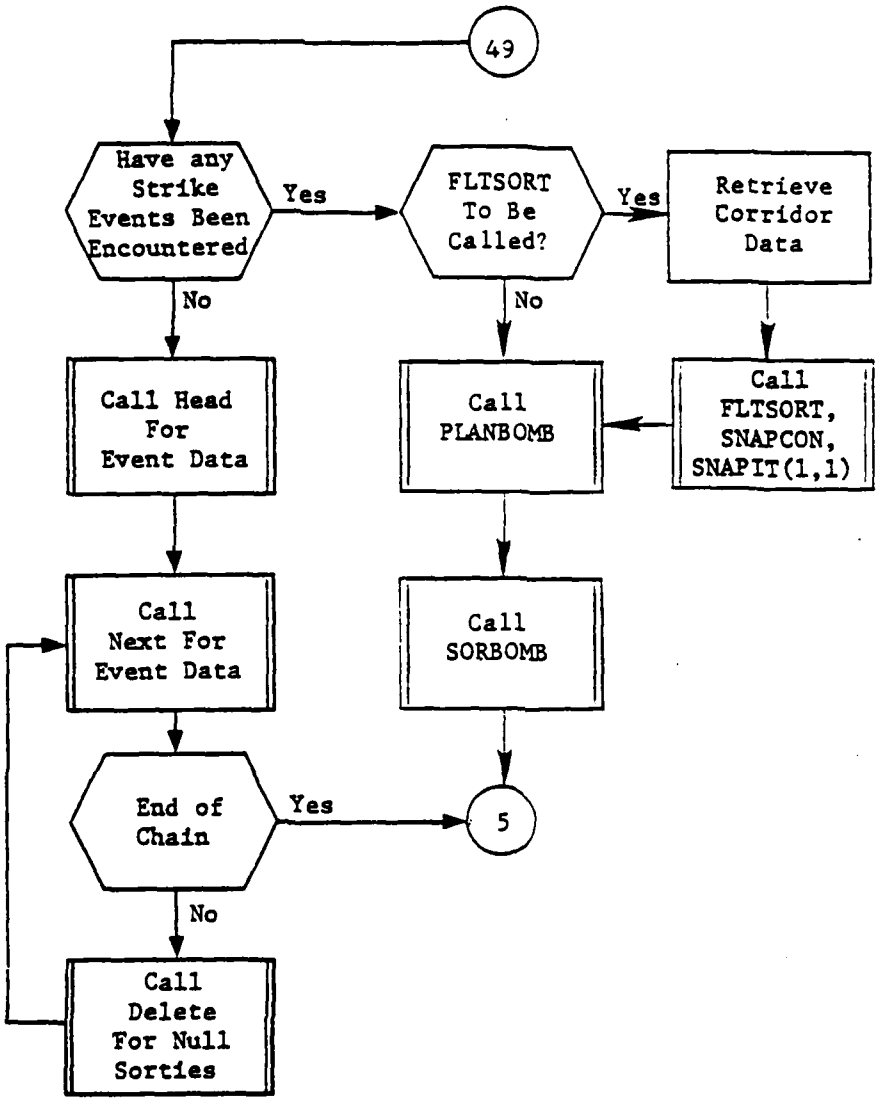


Figure 70. (Part 21 of 35)

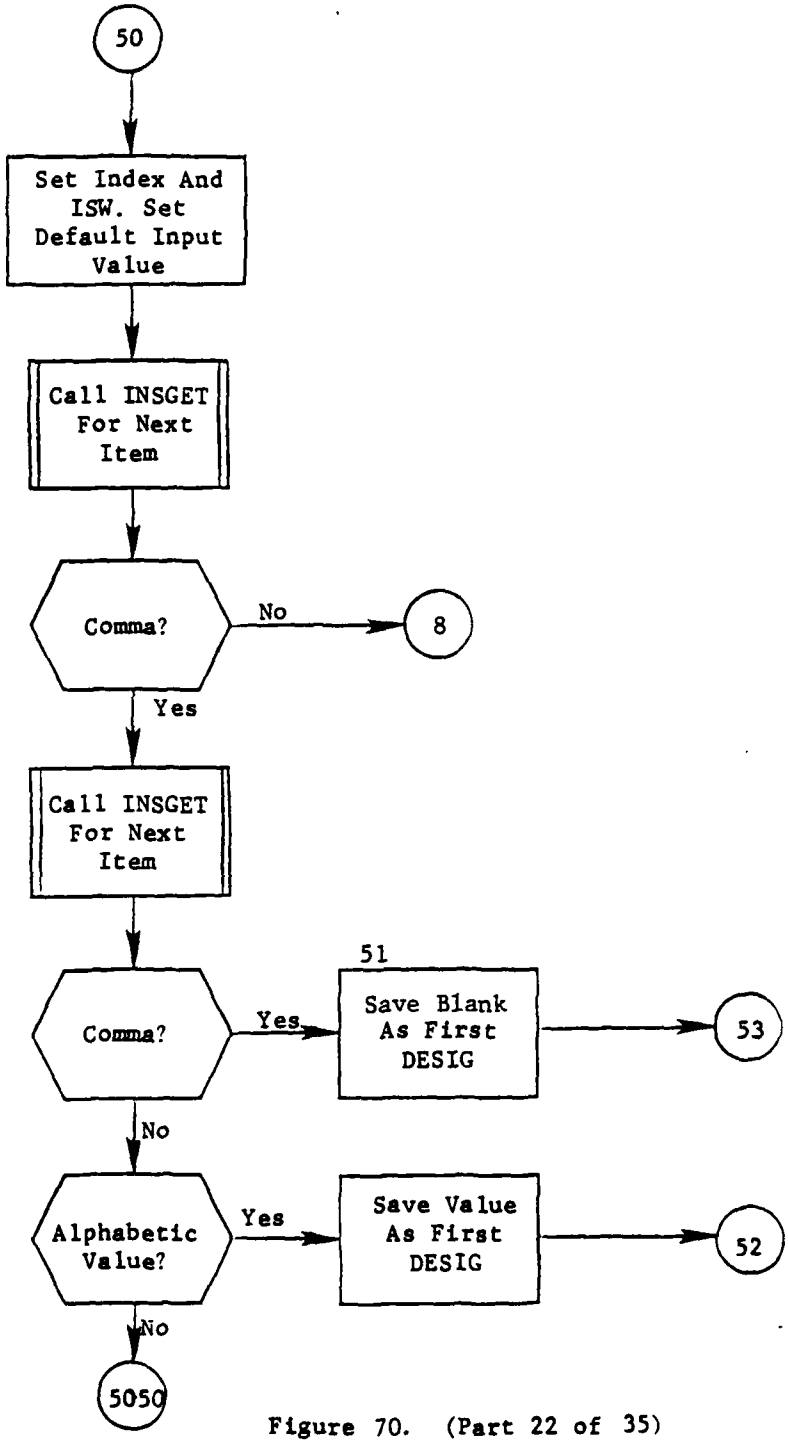


Figure 70. (Part 22 of 35)

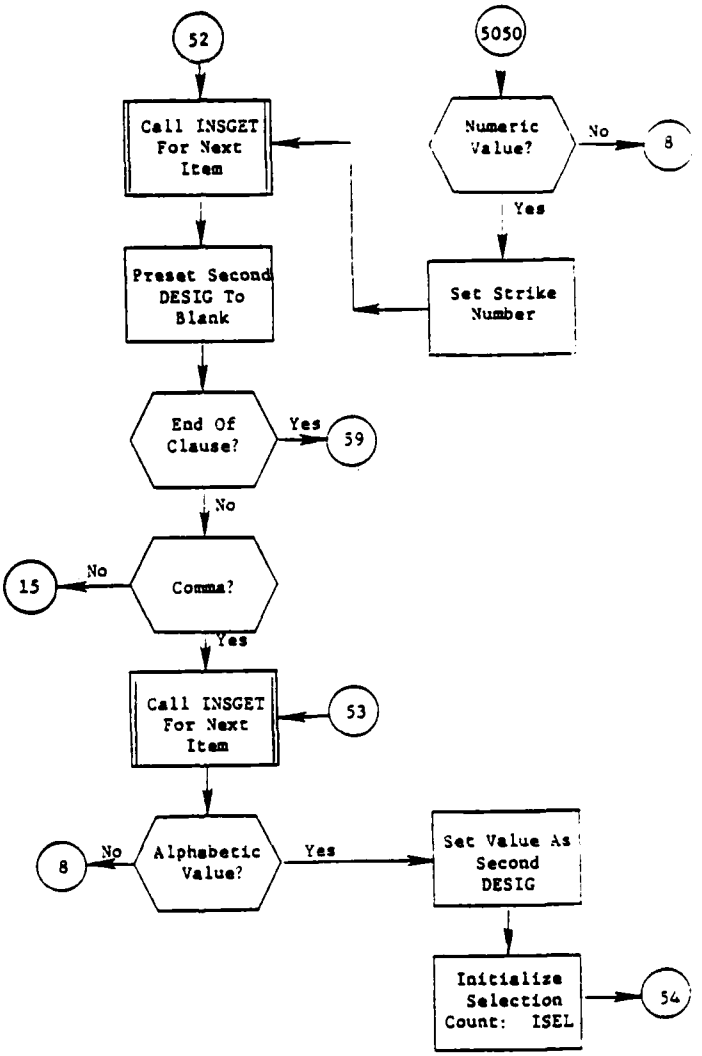


Figure 70. (Part 23 of 35)

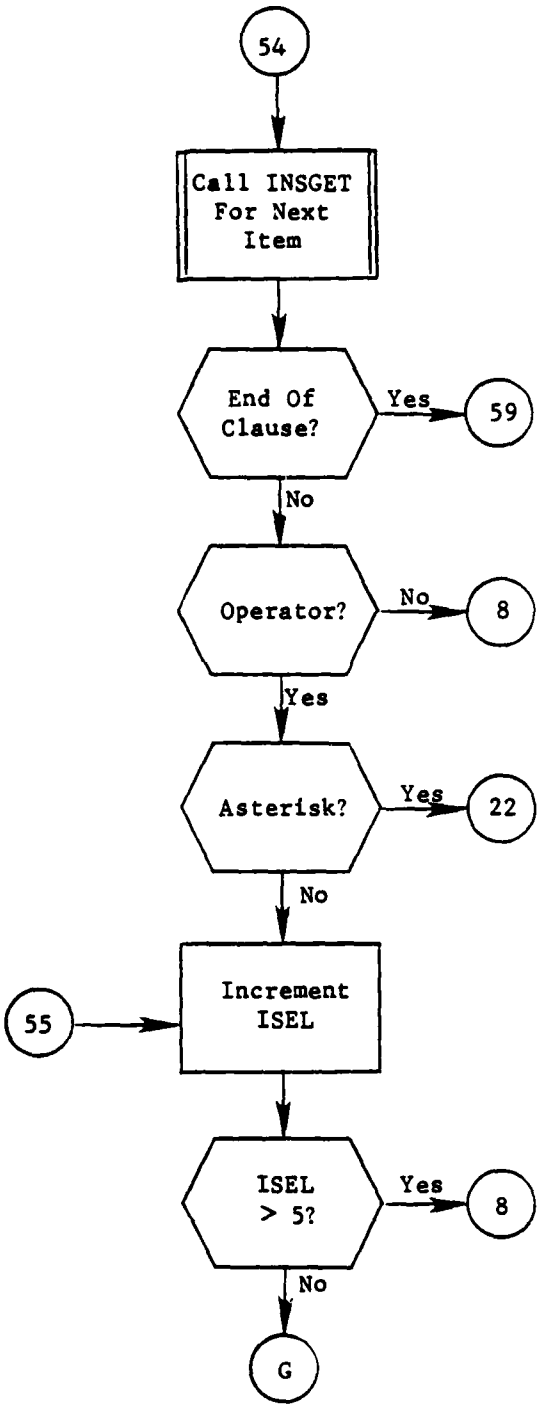


Figure 70. (Part 24 of 35)

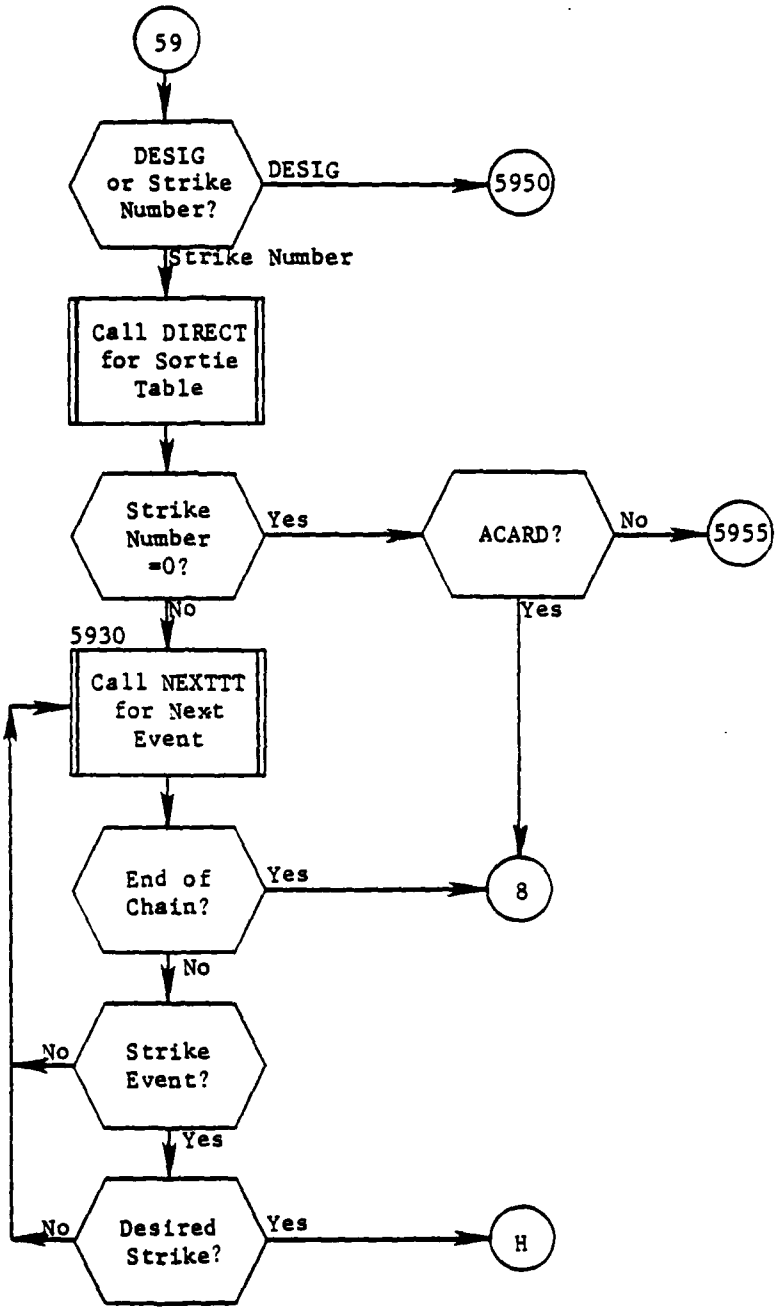


Figure 70. (Part 27 of 35)

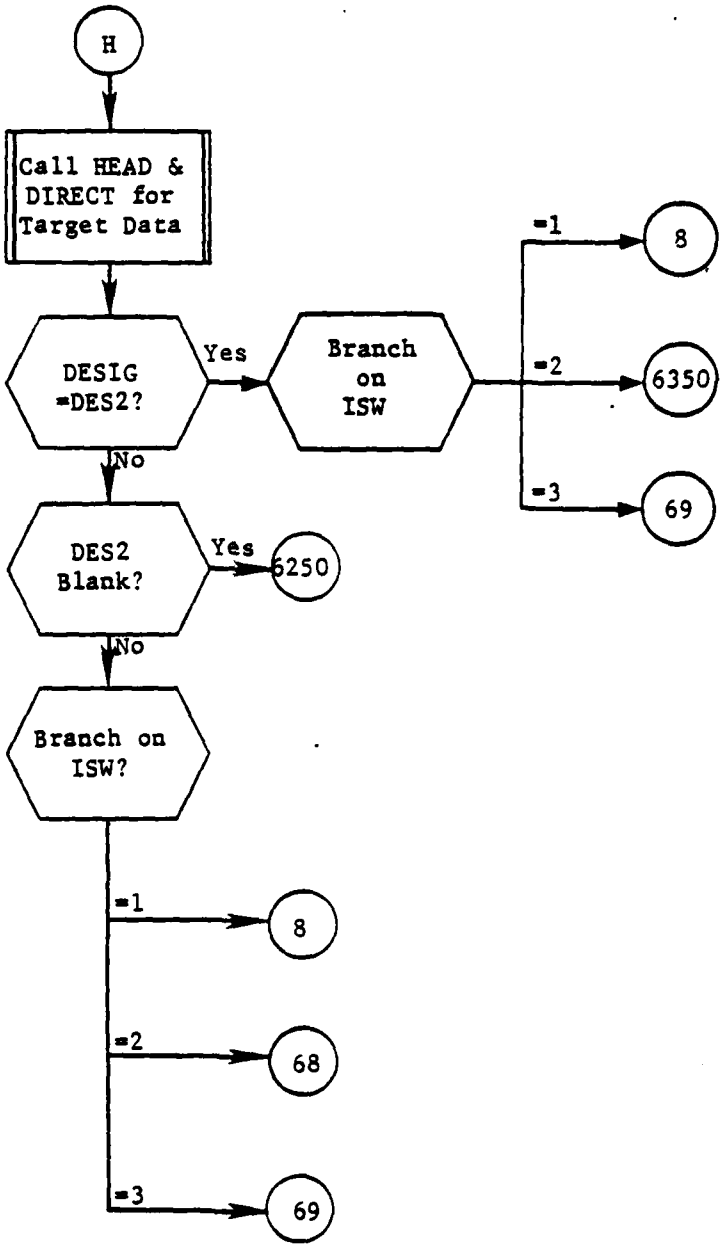


Figure 70. (Part 28 of 35)

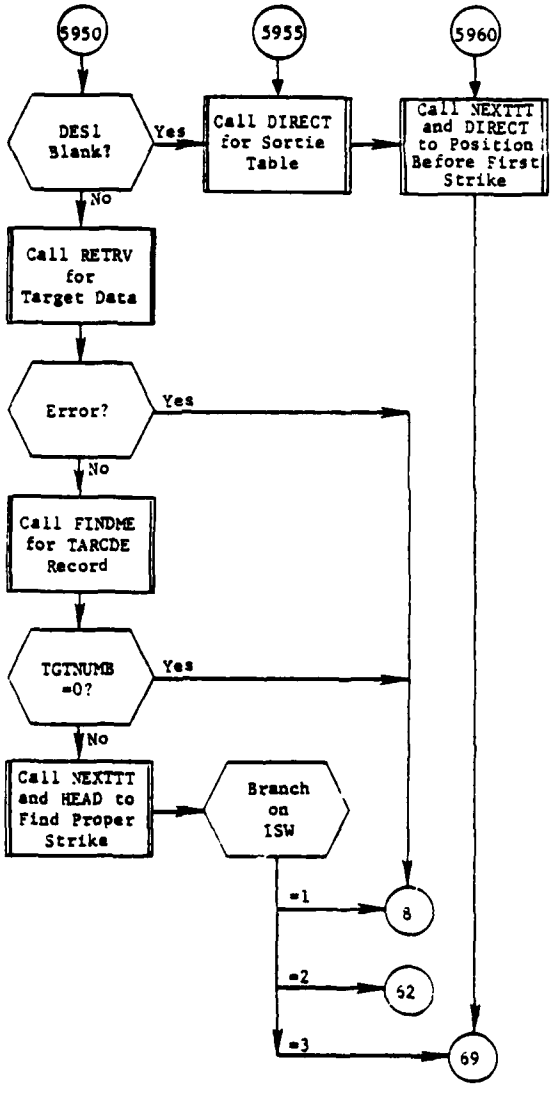


Figure 70. (Part 29 of 35)

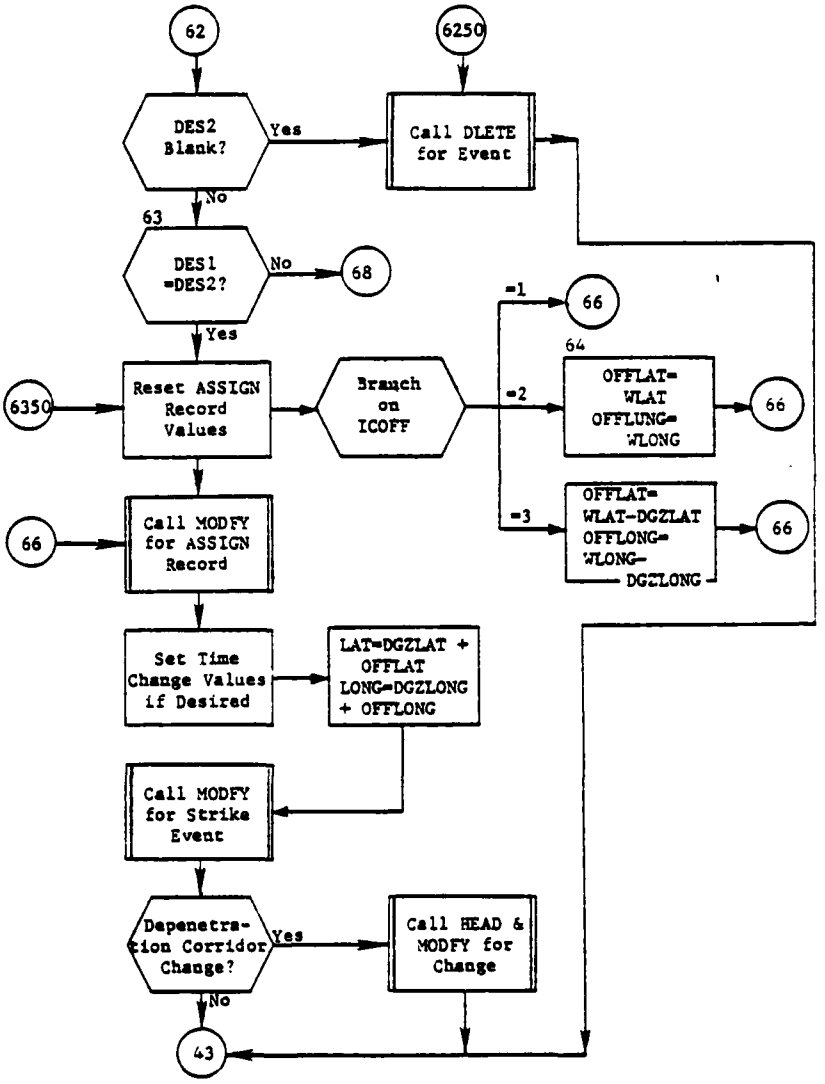


Figure 70. (Part 30 of 35)

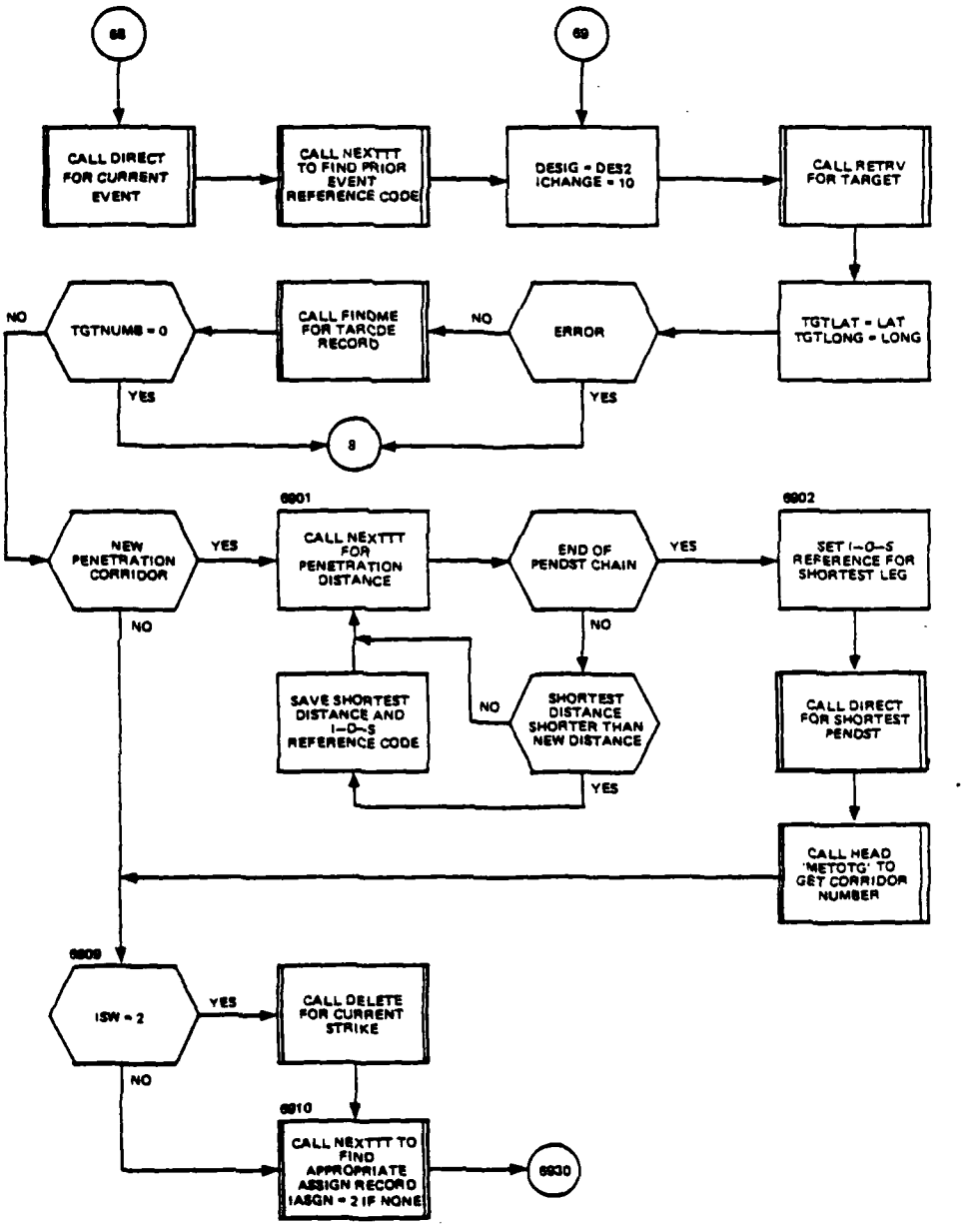


Figure 70. (Part 31 of 35)

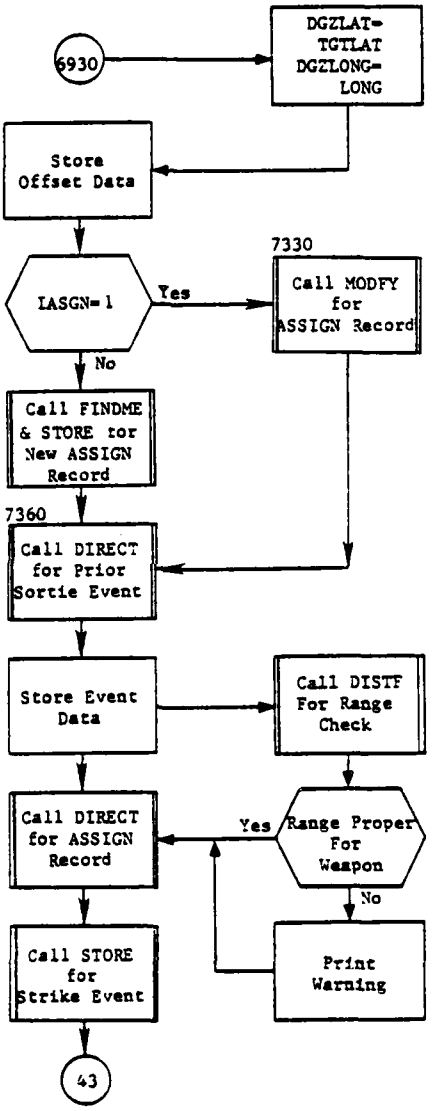


Figure 70. (Part 32 of 35)

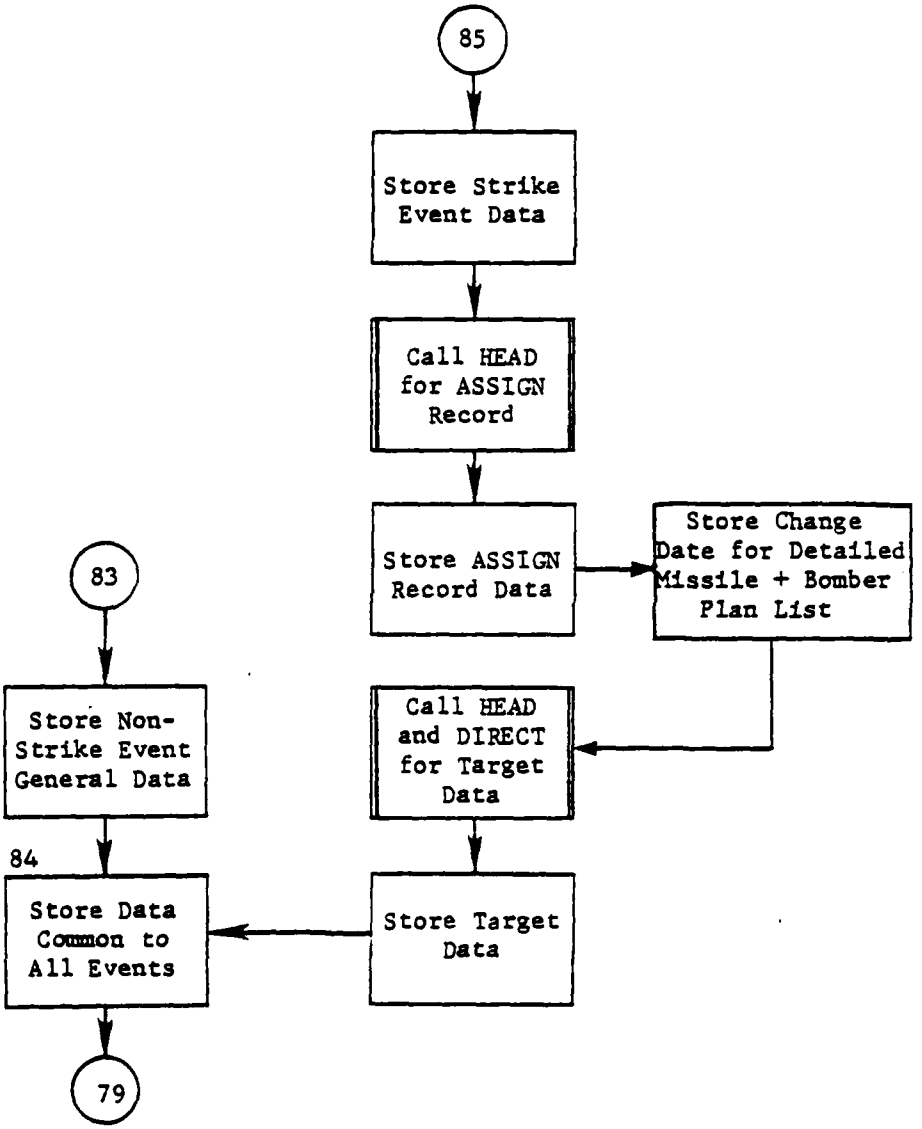


Figure 70. (Part 35 of 35)

4.8.1.1 Subroutine ALTERR

PURPOSE: To report errors in sortie change classes

ENTRY POINTS: ALTERR

FORMAL PARAMETERS:
ICLCNT - Ordinal count of clause in error
KSW - Type of clause: 1 - ACARD
2 - CCARD
3 - ICARD
ICODE - Error message code
JDEX - Index for start of clause

COMMON BLOCKS: None

SUBROUTINES CALLED: INSGET

CALLED BY: ALTPLAN

Method:

The clause in error is rebuilt in CLAUSE by inserting those items found as instructions. After the rebuilding is complete, the reconstructed clause is displayed along with an error message retrieval from the CODE array.

Subroutine ALTERR is illustrated in figure 70.1.

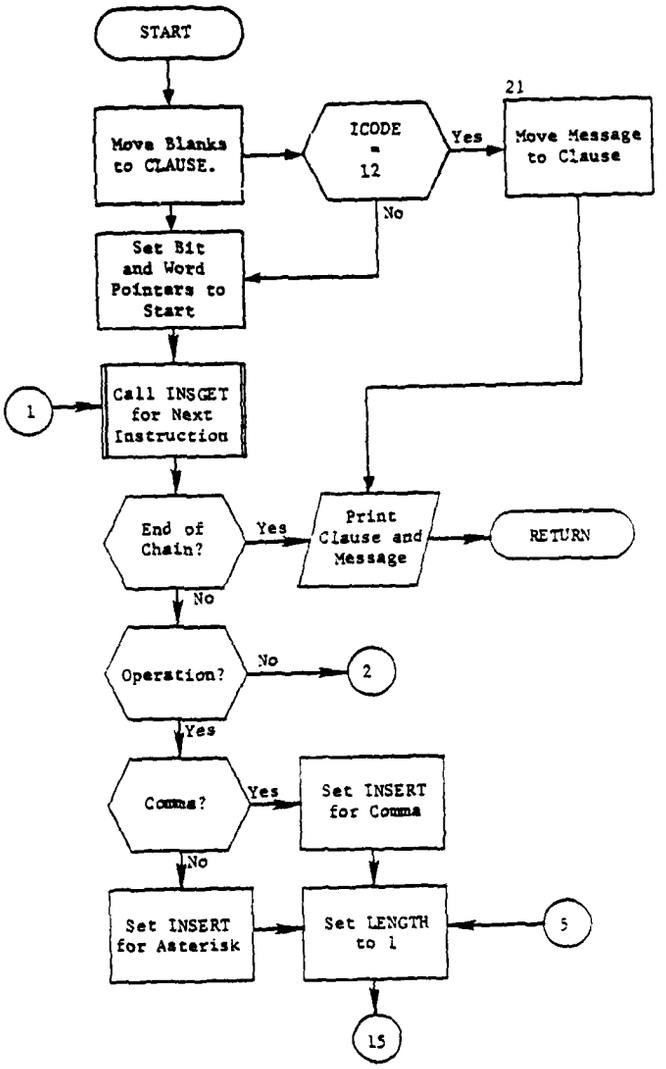


Figure 70.1 Subroutine ALTERR (Part 1 of 2)

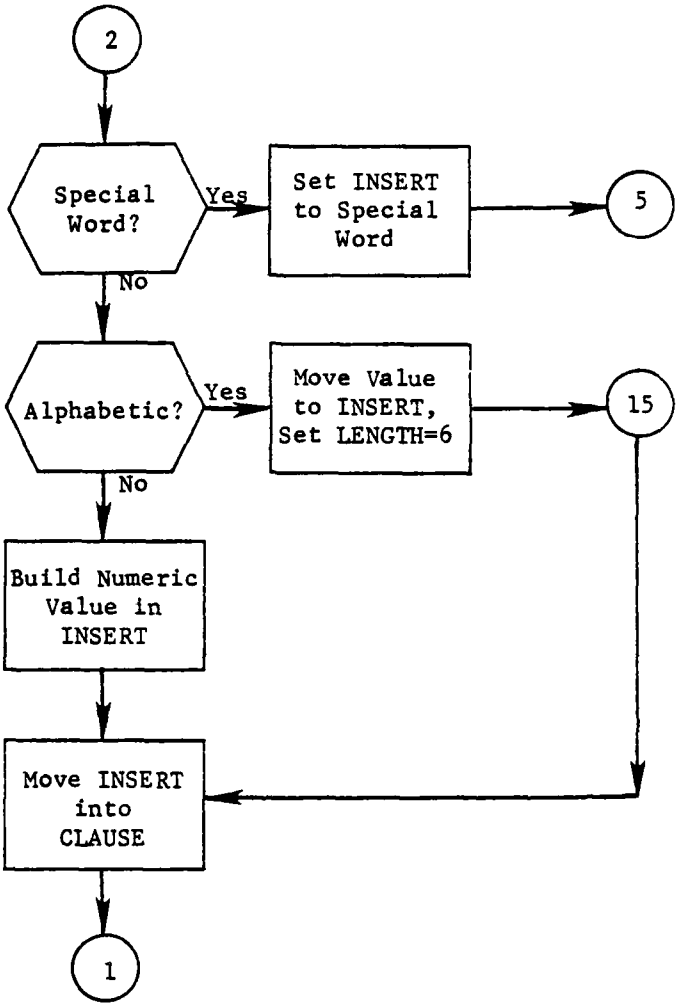


Figure 70.1. (Part 2 of 2)

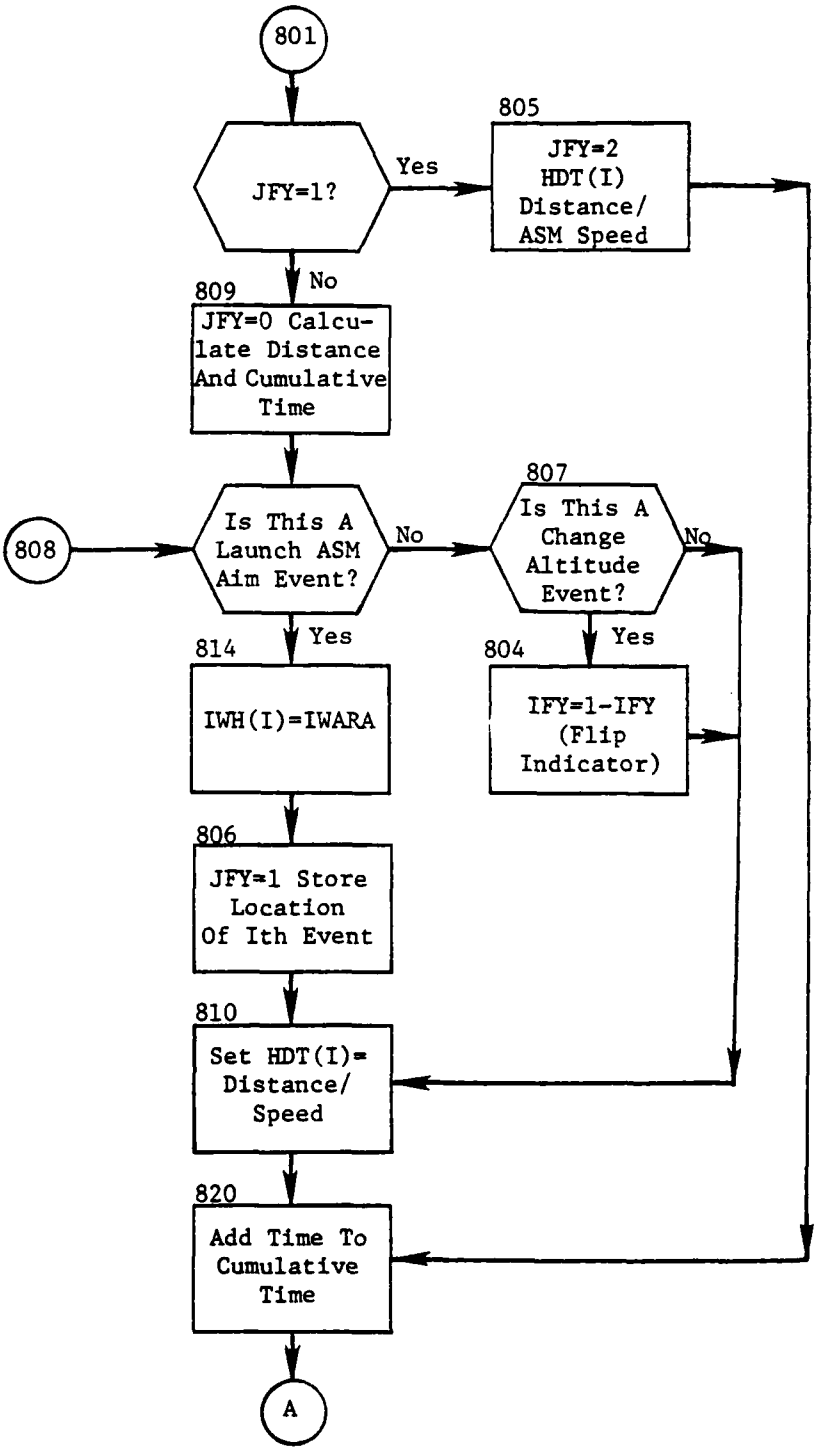


Figure 77. (Part 3 of 3)

4.8.6 Subroutine FINDME

PURPOSE: To find TARCDE, SRTYTB and WEPNGP records during the sortie change process

ENTRY POINTS: FINDME

FORMAL PARAMETERS: INDX - Index for use in search:
TGTNUMB for TARCDE records
SORTNO for SRTYTB records
GROUP for WEPNGP records

ISW - selects record searched for:
1 - TARCDE
2 - SRTYTB
3 - WEPNGP
4 - Indicates initializing call to subroutine

COMMON BLOCKS: C10, C15, C30, LASREF

SUBROUTINES CALLED: DIRECT, HDFND, NEXTTT, RETRV

CALLED BY: ALTPLAN

Method:

Subroutine FINDME is called by ALTPLAN to retrieve various types of records which the user has identified in sortie change clauses. Each of the three record types (TARCDE, SRTYTB, and WEPNGP) reside on a chain (LISTXX, SORTIE, and WEPGRP, respectively) in ascending order of their identifier (TGTNUMB, SORTNO, and GROUP, respectively). However, there is no direct way of retrieving an individual record built into the IDS system other than starting at the chain header and cycling the chain until the desired record is found. FINDME is designed to reduce processing time in cases where a number of such retrievals are called for. It does this by establishing reference points along each chain. These reference points are the IDS reference codes of records on the chains. Then, when a record is called for, FINDME can begin the search at a point on the chain closer to the desired record than the header and thus reduce search time. FINDME is designed to reserve up to 100 reference points for each chain.

For example, let us say that a sortie change clause referred to sortie number 173 out of 300 sorties. Since the total number of sorties is 300, a reference would be established at every third SRTYTB record on the SORTIE chain starting with SORTNO=1. There would, therefore, be a reference for SORTNO=172. This SRTYTB record (SORTNO=172) would be retrieved and the SORTIE chain cycled until the SORTNO=173 record was found. This would reduce the number of calls to NEXTTT from 173 to 1.

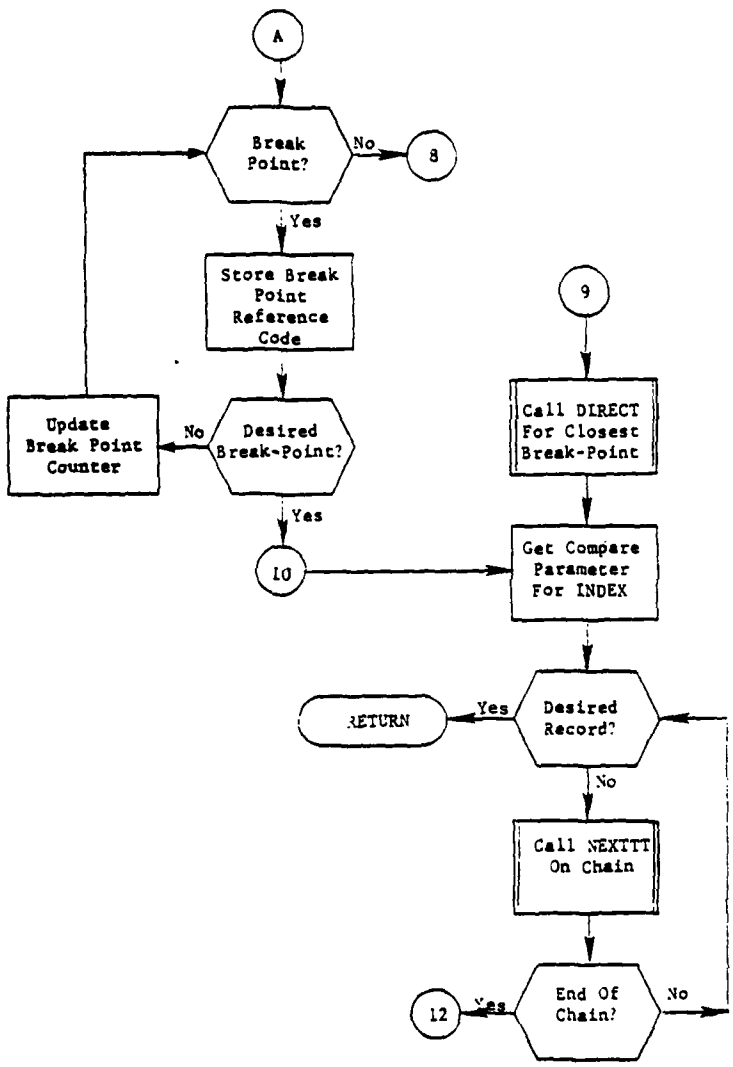


Figure 78. (Part 4 of 4)

4.8.7 Subroutine FLTSORT

PURPOSE: Recalculate bomber attrition and available low altitude range.

ENTRY POINTS: FLTSORT

FORMAL PARAMETERS: None

COMMON BLOCKS: CONTRl, CORRCHAR, CORRCl, DINDATA, DPENREF, GRPSTF, OUTSRA, OUTSRT, PAYSTF, TYPSTF, WHDSTF

SUBROUTINES CALLED: DISTF

CALLED BY: ALTPLAN

Method:

FLTSORT recalculates bomber's survival probability, attrition, and available low-altitude range, if necessary. Certain changes on sortie change cards, such as addition or deletion of targets from the original sortie, require the altered plan be reevaluated. For minor changes, time or offset, the user selects whether recalculation is desired.

The basic logic of subroutine FLTPLAN of module POSTALOC is used for the recalculation. FLTSORT must initialize target values and distances from origin, between targets, and to recovery. If the total distance to be flown exceeds maximum, an error message is printed and processing of this sortie stops. If any time change between targets has been requested, an effective distance is calculated along with a corresponding effective velocity. The effective velocity is checked to ensure it does not change from the base value by more than a given percent. If limits are exceeded, a maximum time change is computed and stored as the value for use. After initialization the logic and mathematical technique is similar to FLTPLAN (entry FINFLT is not used).

Figure 79 illustrates FLTSORT.

The list of tanker bases is then scanned to see whether the buddy refuel point is within range of any of them. If not, the closest tanker base is chosen and a new refuel point is computed by interpolation. This point will lie on the line drawn between the tanker base and the original refuel point in such a way that it will be within range of the tanker base.

The actual time of arrival at the refuel area is computed, using the CORBOMB parameter if the plan is for a first strike. The earliest arrival time in each refuel area is saved for later use when generating tanker plans (array ARTIME). Also saved for tanker scheduling is the arrival time and refuel area for each bomber (array ARVLS).

Block 27: Initialize Plan with Respect to GOLOW Range (figure 94)

The low-altitude range available to the bomber in flying the sortie is specified to PLAN in three separate amounts: the amount during the precorridor legs (G_1), the amount immediately prior to the first target (G_2), and finally, the amount immediately following the first target (G_3).

In block 27, these amounts are examined to make certain that the bomber does not fly low for less than 15 minutes. If $G_1 < 15 * SPDHI$, then G_1 is added to G_2. If $G_2 + G_3 < 15. * SPDHI$, then G_2 and G_3 are set to zero.

If the bomber is a tactical or naval aircraft (denoted by the use of corridor 1 or 2), coding blocks 30 and 31 are skipped.

Block 30: Process Precorridor Legs and Apply GOLOW-1 (figure 95)

The main sortie processing begins then at block 30 with the processing of the precorridor legs. They must be processed in the opposite direction from the bomber flight beginning at the origin and proceeding backward toward the entry. This is because the available low-altitude range (G_1) is measured backward from the corridor origin. Corridor attrition may be associated with the precorridor legs, and low-altitude range is applied against only those corridor sections where the bomber would experience attrition. Any G_1 remaining is added to G_2.

The processing for this block of coding is perhaps best described by referring first to figure 96 which gives an example of precorridor legs in the most complex configuration allowed. It also shows how this this corridor is described to the module in /HAPPEN/. The corridor consists of eight separate doglegs or nine points, and so is described in nine lines in /HAPPEN/. Those doglegs where the bomber would experience listing the location (latitude and longitude) of each dogleg point in order beginning at the corridor origin and proceeding backward toward the corridor entry, as shown in the figure. With each point the distance from the previous point is also noted. If attrition begins at a point, this is noted by entering a 1, 2, or 3 in array JAPTYPE, depending upon whether this is the first, second,

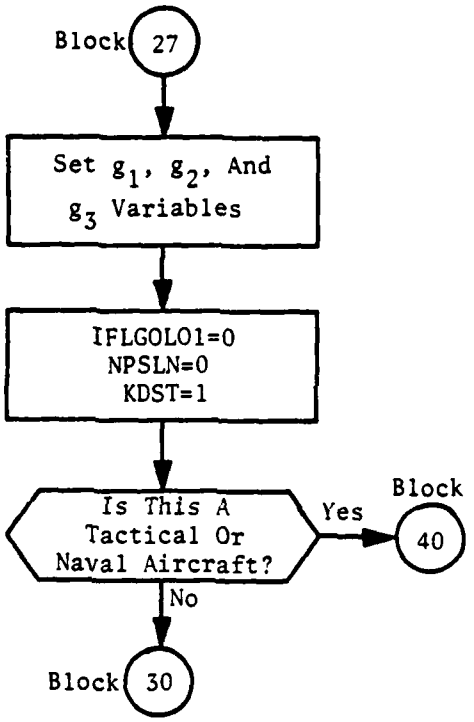


Figure 94. Subroutine PLAN
Block 27: Initialize Plan With Respect to GOLOW Range

4.8.14 Subroutine PLANBOMB

PURPOSE: Control processing of bomber plans

ENTRY POINTS: PLANBOMB

FORMAL PARAMETERS: None

COMMON BLOCKS: ADVRB, C30, DECA, DINDATA, DINDT2, EVENTS, MH2, OUTSRT, POLITE, PPINFO, PPXX, RL

SUBROUTINES CALLED: DECOYADD, DISTF, DISTIME, INTERP, PLAN, SNAPIT, SWTCHALT

CALLED BY: ALTPLAN, PLNTPLAN

Method:

The first action of this subroutine is to call subroutine PLAN. Together these two subroutines control the processing of bomber plans. In the discussion that follows, the two subroutines are to be thought of as sequential parts of an integral process.

Figure 102 shows a typical path a bomber would take between the time of its launch and its recovery. The bomber is launched from a base, flies to a refuel point or area if refueling is called for, then to a corridor entry point. It may then fly one or more prespecified doglegs (called precorridor legs) which define a penetration route before reaching the point labeled Corridor Origin. From the origin it flies over the target area and its assigned targets in their proper order. It then enters the depenetration corridor which may also consist of one or more doglegs. From there it flies to the recovery point or base.

This path may logically be divided into four parts: (1) the launch and refuel portion, (2) the precorridor legs, (3) the target area which is the main part of the plan, and (4) the depenetration and recovery portion.

In PLANBOMB/PLAN, each bomber sortie is processed in much the same order as it is flown; that is, first the precorridor section events are posted, then those of the target section, and finally, the depenetration and recovery section events. Besides the posting of the target events themselves, the main processing consists of posting events for changes of altitude and decoy launches. All postings for bomber events are made in the arrays of common /DINDATA/. The completed plan is output, and processing begins on the new plan.

CH-2

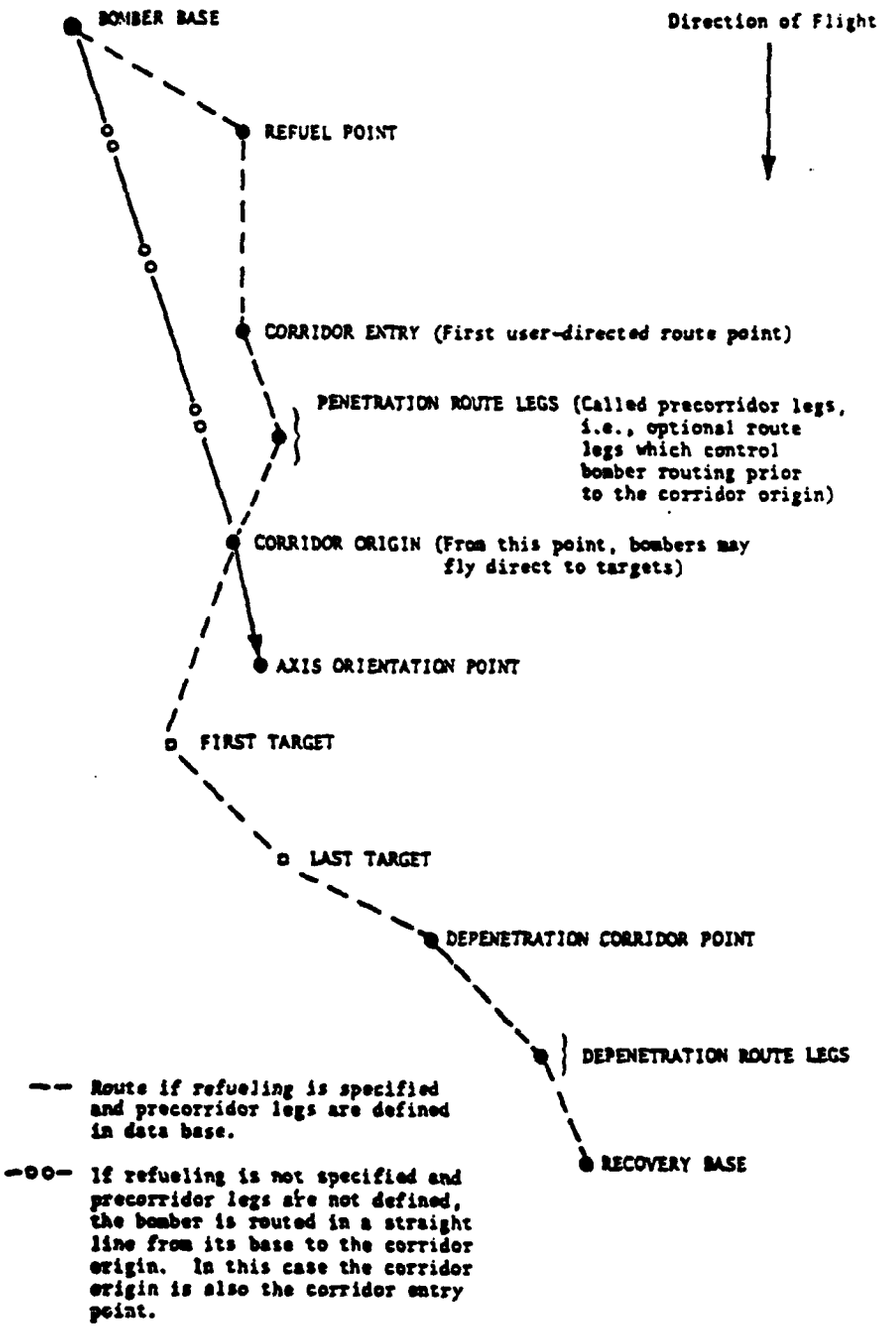


Figure 102. Path of Typical Bomber Sortie

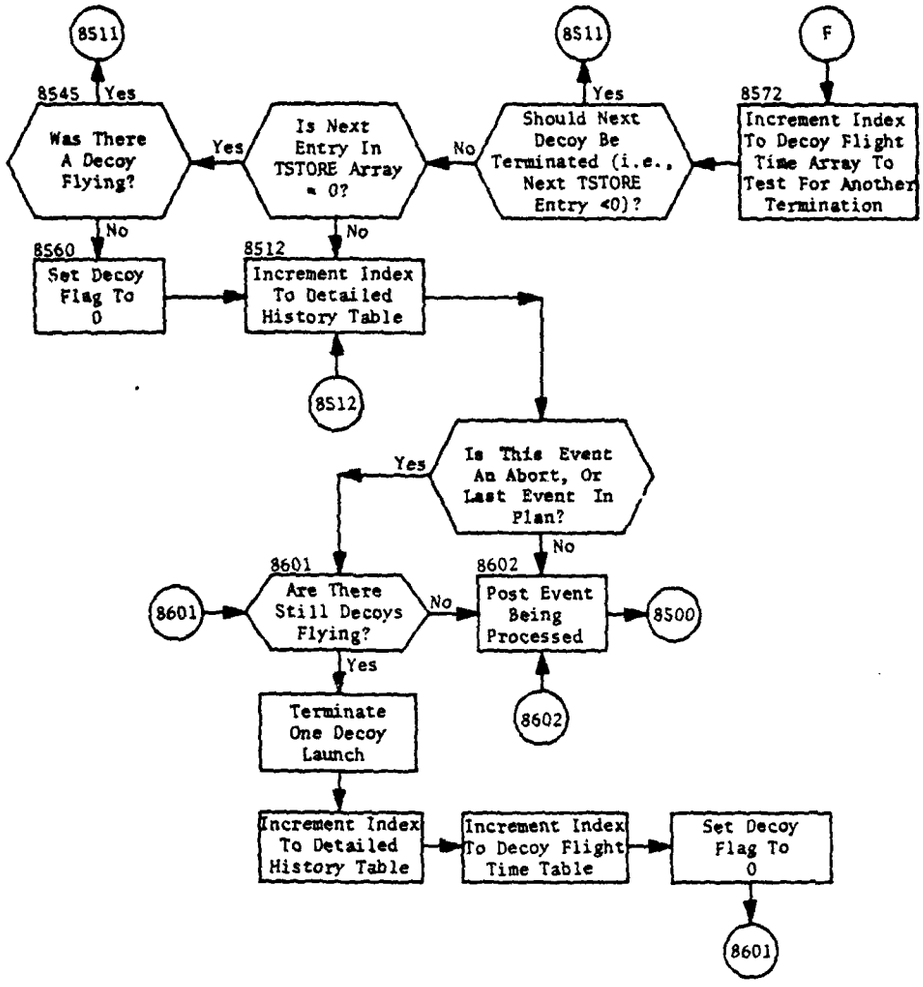


Figure 103. (Part 8 of 9)

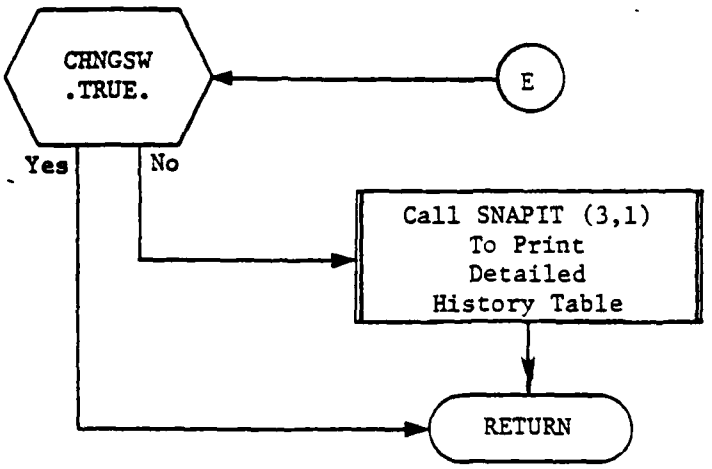


Figure 103. (Part 9 of 9)

4.8.15 Subroutine PLANTMIS

PURPOSE: Control processing of missile plans.

ENTRY POINTS: PLANTMIS

FORMAL PARAMETERS: None

COMMON BLOCKS: ADVRB, C10, C30, GRPSTF, TIMELINE, TYPSTF

SUBROUTINES CALLED: ATN2PI, DISTF, GLOG, HEAD, MODFY, NEXTTT

CALLED BY: ALTPLAN, PLNTPLAN

Method:

The subroutine's main function is to determine the missile launch time based on any timing information provided by the user via MISTME and MSLCOR clauses (subroutine LNCHDATA). First the subroutine checks through all the target assignments. This information is needed since, if there are several targets assigned to a missile and more than one have fixed time assignments, only the first fixed time assignment encountered will be considered. Thus, if a previous fixed time assignment has determined the launch time for the missile, no further calculations need be done to compute the launch time for later reentry vehicles on the missile. If there are no fixed assignments (with timing) on a missile with MIRV payload, the launch time is computed by considering only the data for the target assigned to the first reentry vehicle on the booster.

For salvoed missiles, local parameter DELTA is calculated based on salvo number, number of simultaneous launches, and launch interval. DELTA is the amount added to the basic launch time for launch interval constraints.

If the weapon is not fixed, PLANTMIS checks the plane type. If the strike is retaliatory (INITSTRK = 2) the complicated time plan is ignored and the launch time is the time specified by FOOTPRNT. If INITSTRK = 1 there are two options. If the missile type has a FLIGHT CORMSL the launch time is computed so that the fraction of the flight specified by CORMSL is completed at time zero. If the missile type has a LINE CORMSL the situation is more complex.

The subroutine then calculates whether the missile flight path crosses one of the timing lines input to subroutine LNCHDATA. If the missile crosses a line, the launch time is computed so that the missile crosses the timing line at time equal to CORMSL. If the missile fails to cross any line, the launch time is chosen so that the missile will impact at time zero.

Finally, the launch time is stored in attribute SLOW1 and the sortie table (SRTYTB) modified.

Subroutine PLANTMIS is illustrated in figure 104.

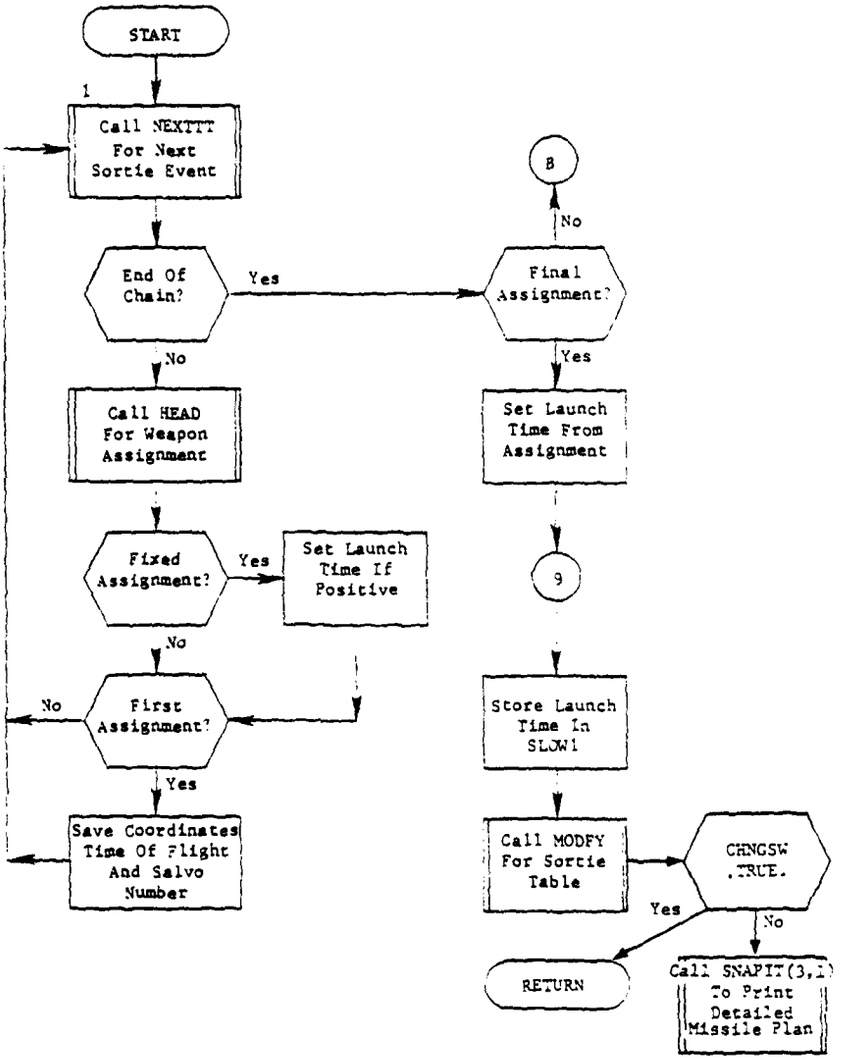


Figure 104. Subroutine PLANTMIS (Part 1 of 3)

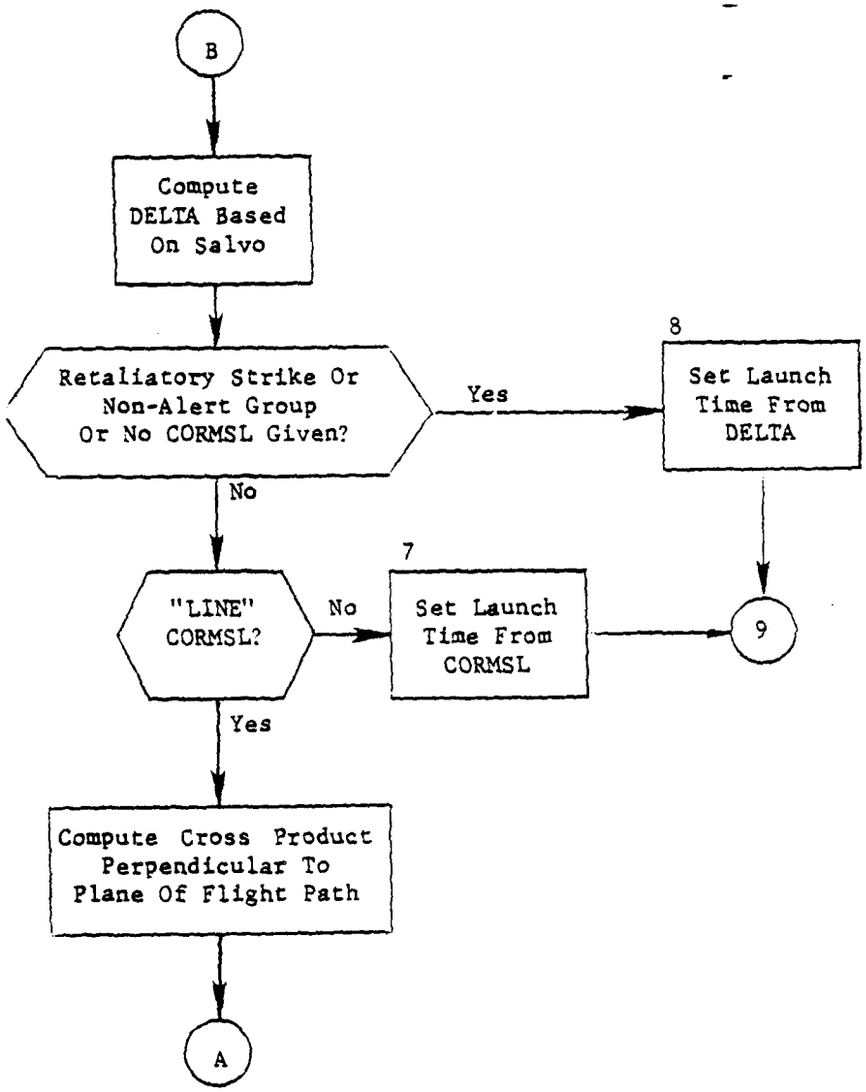


Figure 104. (Part 2 of 3)

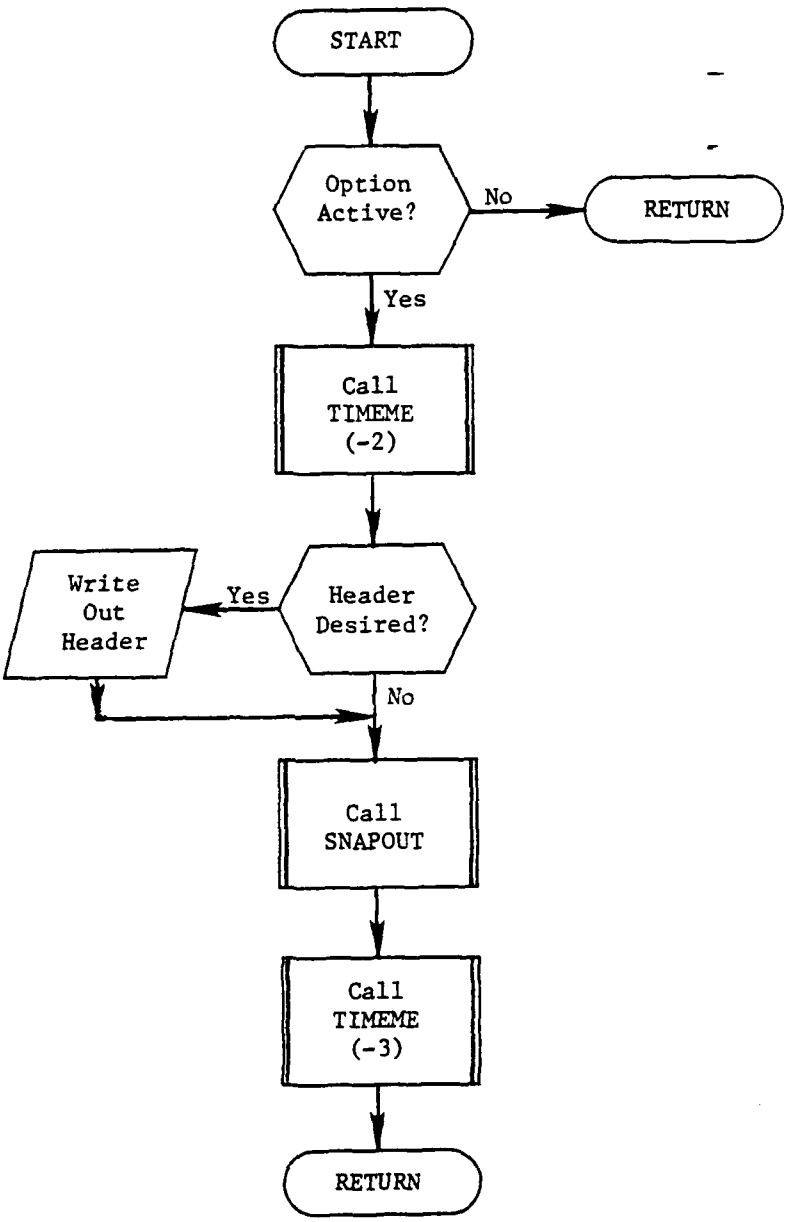


Figure 107. Subroutine SNAPIT

4.8.19 Subroutine SNAPOUT

PURPOSE: To perform all optional printing within PLANOUT.

ENTRY POINTS: SNAPOUT

FORMAL PARAMETERS: ILK = Print request number
MLK = Print option code

COMMON BLOCKS: ASMARRAY, C30, CONTROL, CORCOUNT, DINDATA, DINDT2, DISTC, GRPSTF, HILO, IDP, INDATA, IOUT, IRF, LASM, LAUNSNAP, OUTSRT, OUTSRA, PAYSTF, SPASM, TYPSTF, WHDSTF, DATEOFC

SUBROUTINES CALLED: CONVLL, DISTF, GLOG, TIMEME

CALLED BY: SNAPIT

Method:

The Users Manual (UM 9-77, Volume IV) describes the optional prints available in PLANOUT (numbered 1 through 15), and the data card format to be used when requesting them. The cards are printed initially by subroutine SNAPCON.

Then during processing, subroutine SNAPCON is entered as each new sortie is read in to be processed. SNAPCON scans the list of requests and determines, by matching the group, corridor, and sortie numbers of the incoming sortie against the request list, which prints are to be activated and which are to be inactive during the processing of the record. It communicates this information to subroutine SNAPIT via common block /SNAPON/. This contains a 15-word array NAP, one cell for each print request. The cell is set to 3 for active requests; otherwise it is set to 1.

The subroutine SNAPIT is called wherever a particular print might possibly be issued. For example, SNAPIT is called upon to print the detailed plan immediately after this plan has been completed. SNAPIT then checks cell 3 of NAP and issues the print only if the cell is set to 3. It calls on subroutine SNAPOUT to do the actual printing. This separation of subroutines is made because the resulting FORTRAN-produced program is more efficient; SNAPIT and SNAPOUT might logically be treated as one subroutine.

SNAPOUT itself contains only printing routines. In some instances, the print option code (MLK), passed with the print request number, may select differing print options within the given print number.

Subroutine SNAPOUT is illustrated in figure 108.

4.10 Subroutine INTRFACE*

PURPOSE: Driver subroutine for overlay which creates ABTAPE and STRIKE tapes

ENTRY POINTS: INTRFACE

FORMAL PARAMETERS: None

COMMON BLOCKS: ADVRB, C10, C15, C30, EVCOM, NOFSYS, OOPS, PRNCON, PSW, WEPTRN

SUBROUTINES CALLED: ABOUT, DIRECT, HDFND, HEAD, IFSET, NEXTTT, RDCLAUSE, RETRV, STOUT, WEPDATA

CALLED BY: ENTMOD (PLANOUT)

Method:

First the print switches in block /PSW/ are set according to block /PRNCON/. Also if the frequency for the ABTAPE print option (13) is not 1, the print report code (ABUNIT) is reset to 42. RDCLAUSE and IFSET are now called to read the GAMETIME, FUNCOM and all IF and SETTING clauses. WEPDATA is now called for weapon data.

The sortie header is retrieved and each sortie processed in order. After the last sortie any active tapes have file marks written on them and are rewound. For each sortie the following procedure is followed: TYPFIND is called to set type names and numbers. If ABTAPE is active, ABOUT is called to produce an A-Record.

Then each event of the sortie is retrieved. For each event the data in block /EVCOM/ is updated. If the event is a weapon assignment event and STRIKE tape is active, STOUT is called. For any event if ABTAPE is active ABOUT is called for a B-Record.

Subroutine INTRFACE is illustrated in figure 115.

* First subroutine of overlay INTR.

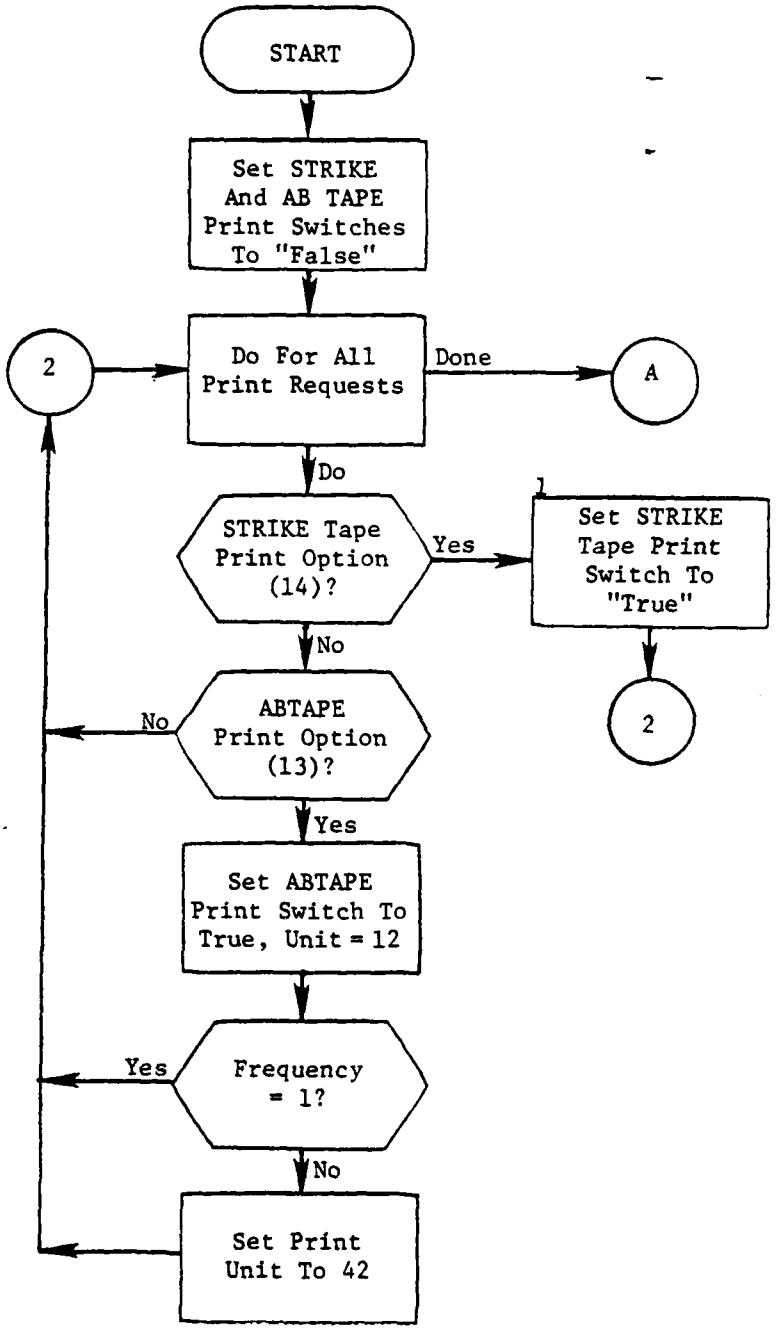


Figure 115. Subroutine INTRFACE (Part 1 of 5)

4.10.1 Subroutine ABOUT

PURPOSE: Write records on ABTAPE.

ENTRY POINTS: ABOUT

FORMAL PARAMETERS: IAB - =1: Produce A-Record
=2: Produce B-Record

COMMON BLOCKS: C30, DEFVAR, EVCOM, GRPSTF, IFSCOM, MODE, PAYSTF, PLTYP, PSW, TYPSTF, WEPTRN, WHDSTF

SUBROUTINES CALLED: CONVLL, FINDTIME, IAZIM, IFUNCT, IGETHOB, INFORM, NOP, NTIME, XSET, XWHERE

CALLED BY: INTRFACE

Method:

First data is stored in variable block /DEFVAR/. Depending on whether the call is for an A or B record, different values are filled. The positions in /DEFVAR/ correspond to record fields as delineated in figure 60. For an A-Record, NTIME is called to set the time of launch. For a B-Record, FINDTIME is called and IAZIM is used for missile azimuth and back-azimuth.

When all data is stored, all pairs of IF and SETTING clauses are executed. Finally, the desired record is formatted using INFORM and written on the output unit (LTN 16).

Subroutine ABOUT is illustrated in figure 116.

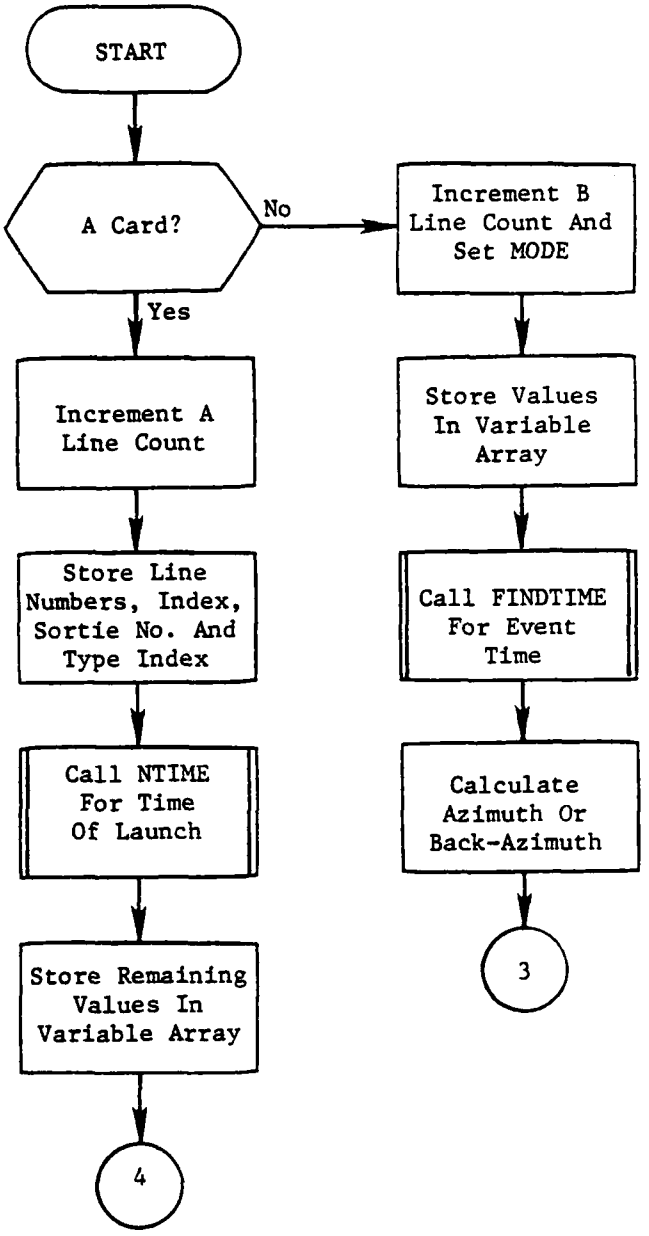


Figure 116. Subroutine ABOUT (Part 1 of 4)

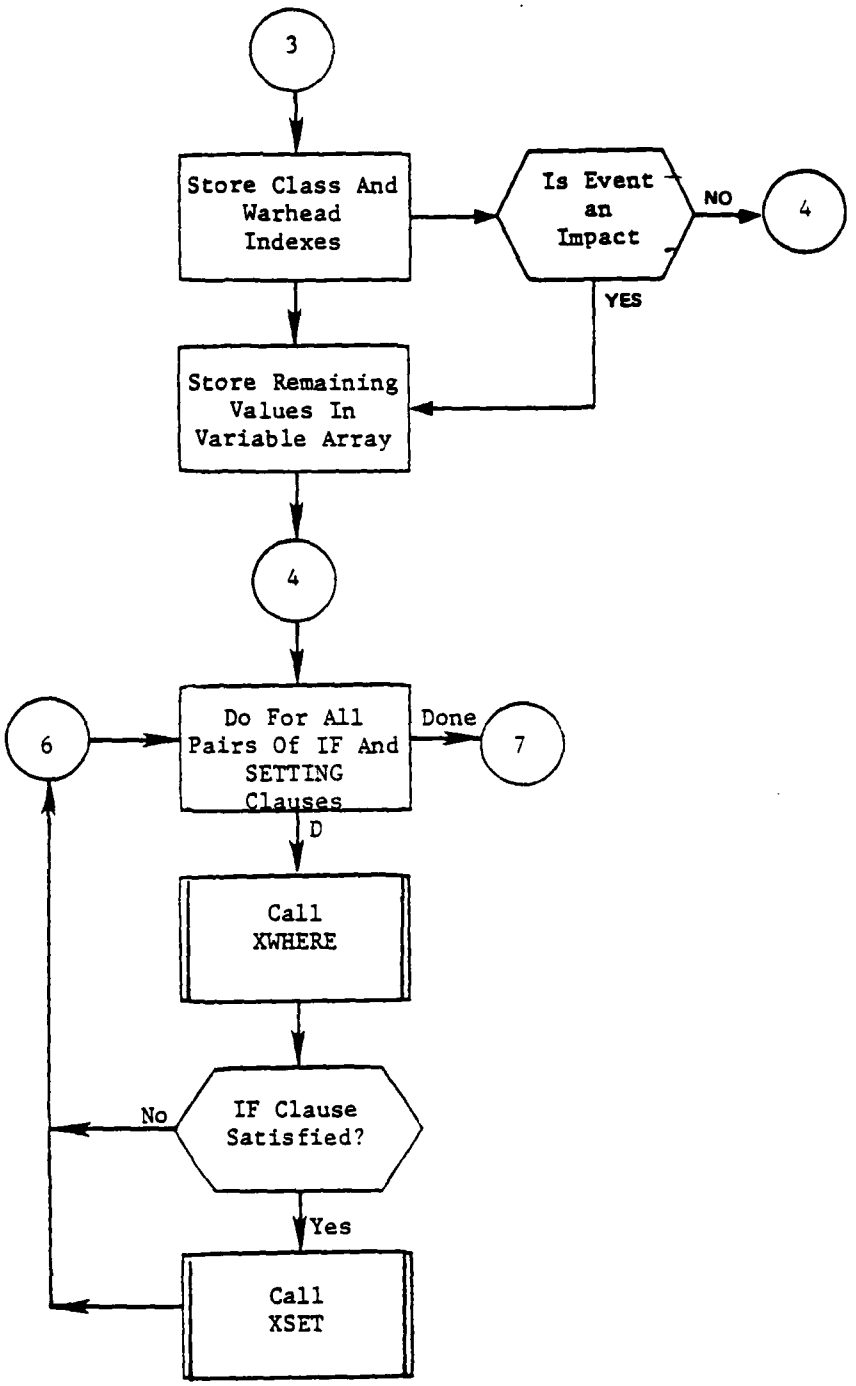


Figure 116. (Part 2 of 4)

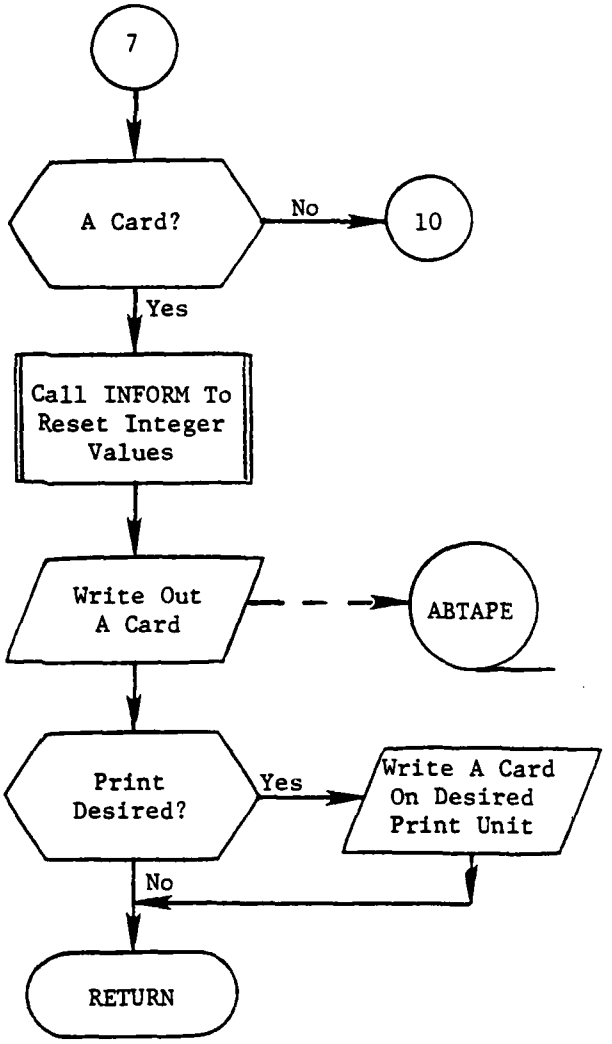


Figure 116. (Part 3 of 4)

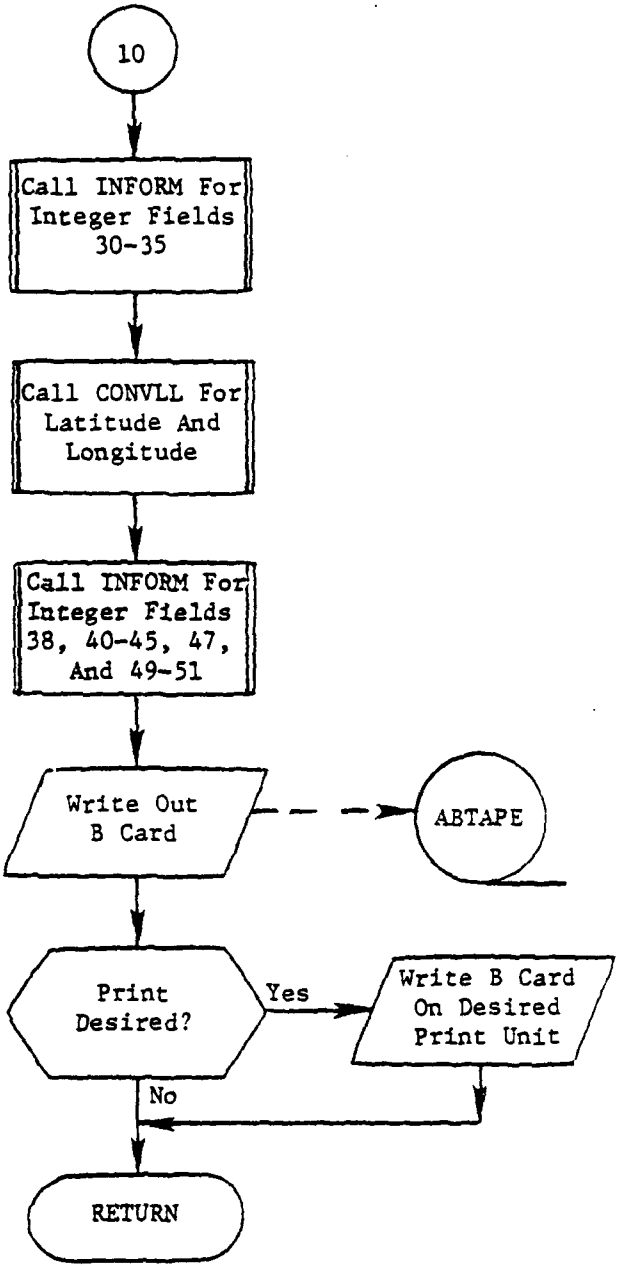


Figure 116. (Part 4 of 4)

4.10.2 Subroutine FINDTIME

PURPOSE: Set times based on input and /GAMETIME/ block. Result is packed into output parameter

ENTRY POINTS: FINDTIME

FORMAL PARAMETERS: XX - Input Time
II - Output time packed as follows
Month * 100000000 + Day * 100000 + Hour * 1000 + Minute * 100 + Second

COMMON BLOCKS: GAMETIME

SUBROUTINES CALLED: None

CALLED BY: ABOUT, STOUT

Method:

Time is set from /GAMETIME/ by adding XX to HHR. Then the subroutine assures that normal limits are observed (i.e., 24 hours/day, etc.) and the result is packed into II.

Subroutine FINDTIME is illustrated in figure 117.

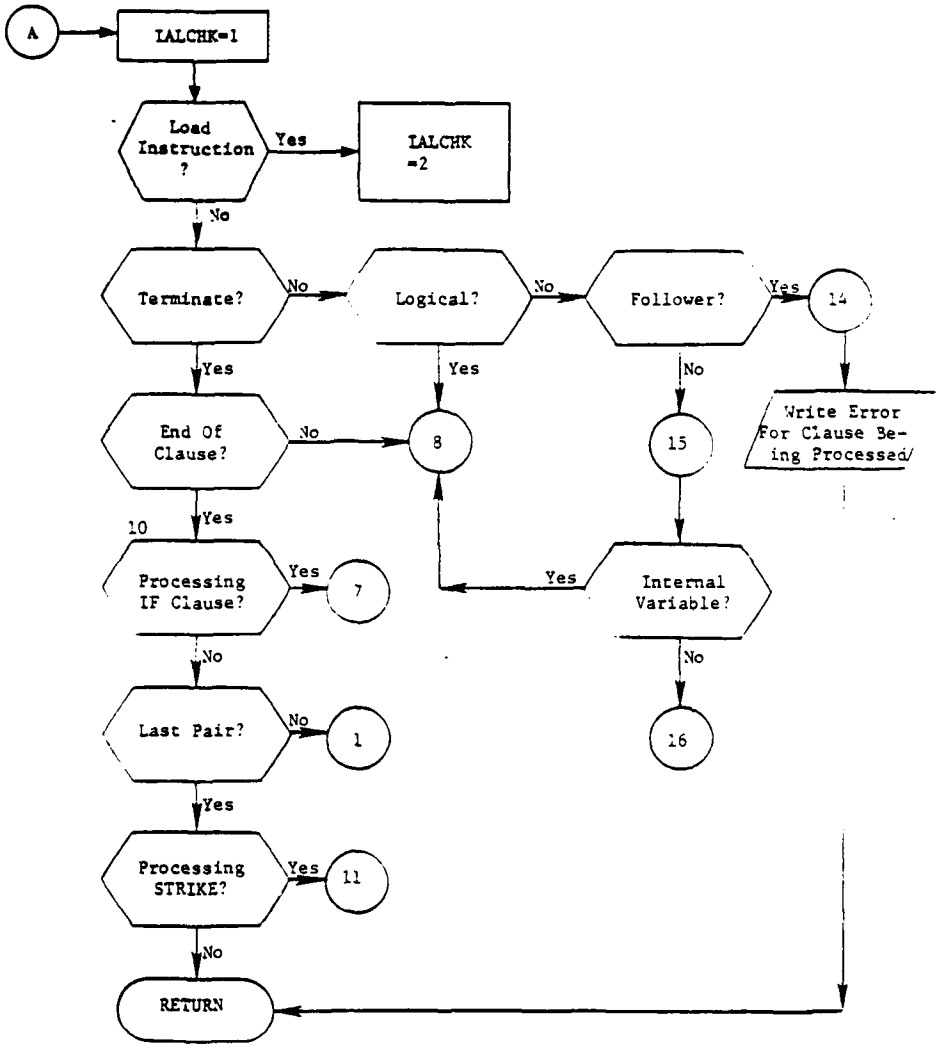


Figure 119. (Part 3 of 4)

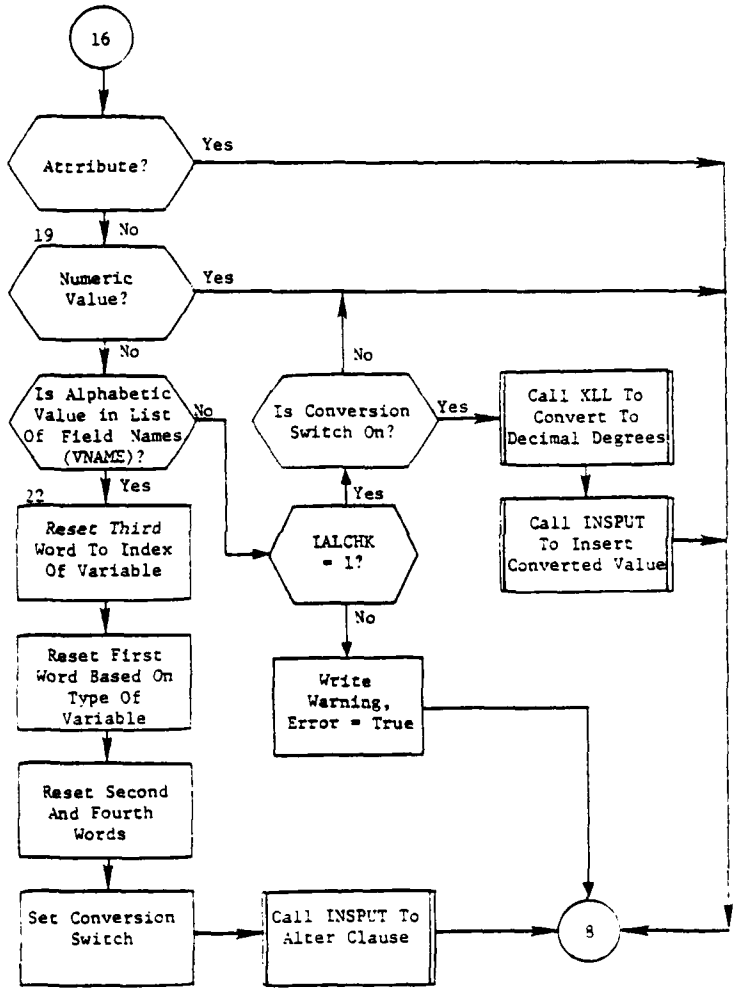


Figure 119. (Part 4 of 4)

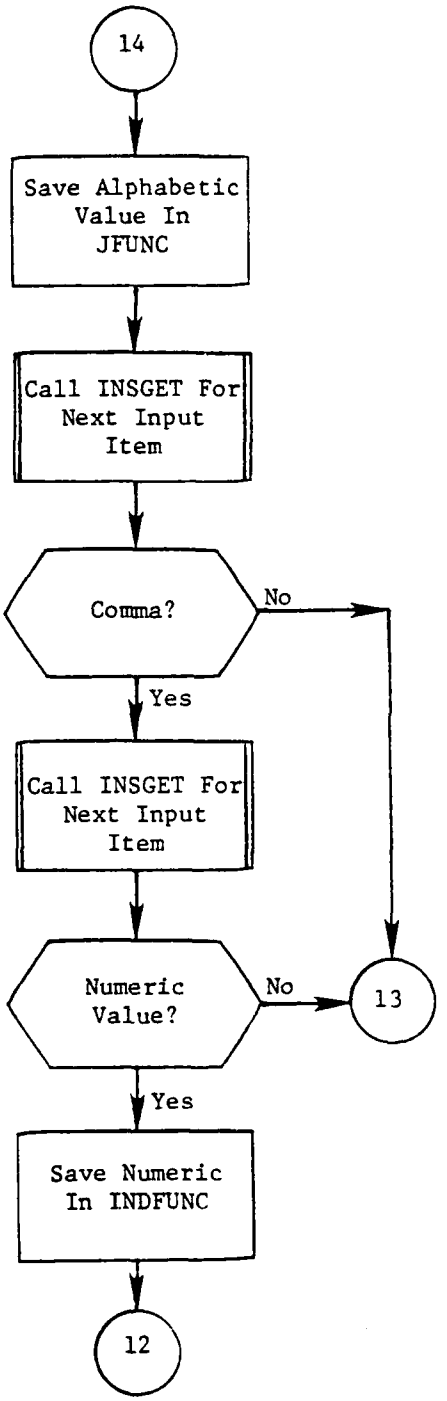


Figure 124. (Part 5 of 5)

AD-A085 635 COMMAND AND CONTROL TECHNICAL CENTER WASHINGTON DC F/G 15/7
THE CCTC QUICK-REACTING GENERAL WAR GAMING SYSTEM (QUICK). VOLU--ETC(U)
JUN 80
UNCLASSIFIED CCTC-CSM-MM-99-77-V4-CH-2 NL

4.10.10 Subroutine STOUT

PURPOSE: Write records on STRIKE tape

ENTRY POINTS: STOUT

FORMAL PARAMETERS: None

COMMON BLOCKS: C30, DEFVAR, EVCOM, GRPSTF, IFSCOM, PAYSTF, PSW, TYPSTF, WEPTRN, WHDSTF

SUBROUTINES CALLED: CONVLL, FINDTIME, IFUNCT, IGETHOB, INFORM, IPROB, XSET, XWHERE

CALLED BY: INTRFACE

Method:

First data is stored in variable block /DEFVAR/. The positions in /DEFVAR/ correspond to record files as in figure 59. When all data is stored, all pairs of IF and SETTING clauses are executed. Finally, the record is reformatted using INFORM and written on the output unit (LTN4).

Subroutine STOUT is illustrated in figure 125.

DISTRIBUTION

Addressee	Copies
CCTC Codes	
C124 (Reference and Record Set)	3
C124 (Stock)	6
C126	2
C313	1
C314	7
C630	1
DCA Code	
205	1
EXTERNAL	
Chief, Studies, Analysis and Gaming Agency, OJCS ATTN: SFD, Room 1D935, Pentagon, Washington, DC 20301	2
Chief of Naval Operations, ATTN: OP-654C, Room BE781 Pentagon, Washington, DC 20350	2
Commander-in-Chief, North American Air Defense Command ATTN: NPXYA, Ent Air Force Base, CO 80912	2
U.S. Air Force Weapons Laboratory (AFSC) ATTN: AFWL/SUL (Technical Library), Kirtland Air Force Base, NM 87117	1
Director, Strategic Target Planning, ATTN: (JPS), Offutt Air Force Base, NE 68113	2
Defense Technical Information Center, Cameron Station, Alexandria, VA 22314	12
	42

THIS PAGE INTENTIONALLY LEFT BLANK